An ECN Probe-Based Connection Acceptance Control

Tom Kelly
Laboratory for Communication Engineering
Cambridge University Engineering Department
Trumpington Street
Cambridge, CB2 1PZ, United Kingdom
ctk21@cam.ac.uk

ABSTRACT

Connection acceptance control is a mechanism which can be used to moderate the load placed on a network by turning away connection requests during times of overload. Traditionally these mechanisms have been implemented by setting up state within a network using signalling protocols, such as the IETF's Resource Reservation Protocol. There have been recent proposals for admission control based on end-systems probing the network to infer network load. These distributed algorithms often have less router state and hence scale more easily. This paper discusses an ECN (Explicit Congestion Notification) probe-based admission control protocol that is fast, scalable and robust. The protocol's performance is then studied through simulation. It is concluded that probe-based admission control is viable in both partitioned and integrated networks, but more research is needed to understand the implications for network policy control.

1. INTRODUCTION

It has been the goal of much research in the past decade to bring about a merging of telephone and computer networks. Such an integrated service network would deliver many benefits in terms of ubiquitous access, economies of scale, improved statistical multiplexing, and new innovative applications. The IETF has put forward two proposals for achieving this aim in their IntServ [9] and DiffServ [7] proposals. The IntServ proposal uses an explicit signalling protocol to allocate and guarantee resources along each route. In contrast the DiffServ architecture provides differentiated service classes which have different per-hop behaviours once inside the network. Implicit in both schemes are the use of connection acceptance protocols to ensure that guarantees can be met and that per-hop behaviours remain predictable.

An ECN-aware router can set the ECN CE bit [16] while experiencing congestion before becoming overloaded and dropping packets. In this way, a receiving end-system can determine if congestion was experienced on the route taken by a packet. It has been suggested that running a low loss and low delay Internet could be possible using ECN marking and appropriate end-system control responses to marks [21]. Such an improvement to the best-effort service model would enable the Internet to support a wider range of applications and might allow an evolutionary path towards an integrated service network. To achieve the aim of a low loss and low delay network requires appropriate congestion control to be used by end-systems that send traffic into the network. For adaptive traffic sources, such as TCP, each flow adjusts its sending rate continually by probing the network to infer the level of congestion. For non-adaptive traffic sources, such as some voice over IP codecs, connection acceptance control algorithms are needed to moderate the total number of non-adaptive traffic sources sending traffic into the network. The connection acceptance control of non-adaptive traffic sources would effectively make the class of non-adaptive flows adjust its aggregate sending rate in accordance with congestion levels. This paper aims to investigate the viability of such an ECN-aware low loss and low delay network supporting non-adaptive real-time traffic.

The use of ECN feedback for elastic rate traffic, such as TCP, is the subject of much research [3, 14, 18, 23, 25]. One TCP response scheme for ECN is currently in the process of being standardized and deployed [16]. This paper presents an ECN probe-based connection acceptance control scheme for real-time non-adaptive traffic. It considers the viability of the scheme in both a network used only for connection acceptance controlled traffic and in an IP network carrying both adaptive and non-adaptive traffic. The viability of the protocol to provide accepted real-time non-adaptive flows with an adequate loss and delay performance is then studied through simulation. This work builds directly on the proposals and mathematical theory behind the distributed acceptance control schemes discussed in [17, 20].

In contrast to traditional schemes, probe-based connection acceptance control distributes the admission decision between the routers within the network and the end-systems utilising the network. Each end-system wishing to initiate a non-adaptive flow would conduct a probe by sending ECN capable packets to the receiving end-system which returns acknowledgement packets indicating whether congestion was experienced by the probe packet. The sending end-system is thus able to infer the level of congestion on the route and perform an admission control decision locally. Such a distributed scheme is appealing because it reduces the network infrastructure costs associated with explicit signalling both between autonomous systems and between the network and end-system. Furthermore distributed probe-based schemes can scale more easily than schemes which require the network to store per-flow state in routers.

The use of probe-based admission control has been previously studied in the context of a DiffServ load controlled service [6, 12]. In contrast the probe protocol presented here is designed to be deployable in a looser environment in which prior topology knowledge is minimal and router ECN support is the only prerequisite. An initial survey and comparison of various endpoint admission control schemes was presented in [11]. This paper builds on some of the conclusions of that work, but attempts to have a more aggressive connection setup time and takes a different approach towards avoiding congestion caused by probes.

This paper is organised as follows. Section 2 discusses the nature of real-time traffic and what is necessary to effectively support it in a computer network. Section 3 introduces a virtual queue marking scheme for marking packets at routers. Section 4 presents the probing protocol used by end-systems to perform connection acceptance control. In Section 5 the method used for the performance simulations is presented. The simulation results and analysis are presented in Section 6. Limitations and directions for further work are discussed in Section 7. Finally Section 8 gives the conclusions to be drawn from the work.

The simulations in this paper were carried out using the ns-2 simulator available at [1].

2. SUPPORTING REAL-TIME TRAFFIC

There are many types of real-time application in which timely delivery of data by the network is important. Examples include video playback on demand, voice conversation, network computer games, and remote surgery. In all these applications there is a notion of a maximum acceptable end-to-end latency. Interactive applications such as voice conversations have some of the most stringent delay requirements and the maximum tolerable round trip delay is in the region of 200-300ms (i.e. an end-to-end latency of 100-150ms).

Network delay is composed of serialisation delay at each router, propagation delay between routers, and queuing delay at each router. For a given route through the network the serialisation and propagation delay are static, while the queueing delay is variable[1]. Queueing occurs when instantaneous demand for a link exceeds the capacity of that link. Queueing delay is thus either caused by bursts or by the prolonged presence of excess offered load on a link.

Despite the highly variable delays and drop rates present in the current Internet, real-time applications can still operate provided they can degrade gracefully in the presence of excess network delay and drops. Significant progress has been made in this area with adaptive buffering and playback techniques able to mitigate queueing delay variability, while retransmission is sometimes viable when packet loss occurs[2]. However such techniques remain inappropriate for some real-time applications due to the latency they introduce when queueing delay is highly variable and drop rates high. To support a rich variety of real-time applications it is thus necessary to provide small queueing delays and low loss rates.

In this paper we consider interactive voice conversations as the real-time application to be supported. Voice is chosen as it is well characterised and is one of the most important interactive applications used in today's communication networks. It should be noted that the probing protocol is also intended to be applicable to other real-time applications. The viability of the ECN probe-based admission control scheme is determined by its ability to provide an accepted call[3] with a low queueing delay and loss rate.

3. ROUTER BEHAVIOR

At present Internet end-systems use the detection of packet loss as the signal of congestion for closed loop congestion control. This means that an end-system can only discover congestion when router buffers are filled to the point of loss. The signalling of congestion can be decoupled from packet drops and queue size by using explicit congestion notification mechanisms. Hence it becomes interesting to ask whether a low delay and low loss network can be constructed with appropriate closed loop control. Appendix A contains some simulation results illustrating some queueing delay and loss characteristics of both an ECN enabled IP network and a standard IP network.

ECN based congestion control schemes require, in comparison to the IntServ and DiffServ proposals, relatively minor alterations to router scheduling and queueing functionality. The use of additional explicit network signalling protocols between host and network is also unnecessary. Routers are only required to detect congestion before the onset of buffer overflow and set the ECN CE bit of the IP packets passing through while congested. Indeed this paper aims to explore what can be acheived with simple FIFO scheduling and appropriate congestion detection embodied in the ECN marking algorithm.

In this study a virtual queue marking scheme as presented in [18] was used as the ECN marking algorithm at routers. Such a scheme was chosen as it has an intuitive feel, is easy to implement, and robust to a variety of traffic types. It is not argued here that this is the optimal marking algorithm and good marking algorithms are the subject of current research [3, 18, 24, 26].

The particular virtual buffer scheme implemented was as follows. Suppose an output buffer of size B is attached to a link of rate C. A virtual queue is then constructed with buffer size θB and drain rate θC, where $0 < \theta < 1$. Packets to be sent on the link are queued in the real buffer B which is serviced at rate C. At the same time the virtual buffer is filled with packets of the same logical size and then drained at rate θC. If the virtual buffer overflows all packets leaving the real queue have the ECN bit set in the IP header until the virtual buffer is next empty[4].

The setting of θ and B is highly dependent on network policy, but a qualitative intuition for the virtual queue's operation is useful. Roughly speaking both θ and B affect the sensitivity of the virtual queue algorithm to detecting congestion. The buffer size B determines the ability of the underlying buffer to accept bursts without loss and also the maximum queueing delay. A large value of B leads to congestion only being detected once queueing delay has

[1] It is assumed that routes between hosts are static on the timescale of connections.

[2] Coding techniques, such as forward error correction, can also make applications resilient to packet loss. This is at the expense of additional implementation complexity and an increase in the rate required to maintain a given quality. Note that an increased use of forward error correction in response to loss exacerbates congestion in a network.

[3] In this paper the terms "call" and "non-adaptive flow" will be used interchangeably.

[4] Packets which do not have the ECN capable bit of the IP header set are discarded if they were to be marked. This does not affect this study as all connections are ECN-aware, but has relevance for the incremental deployment of ECN schemes.

risen dramatically, while a small value of B will limit the resource's ability to separate transient bursts from prolonged congestion. The parameter θ determines the aggressiveness with which congestion is detected. A high value of θ can lead to a higher utilisation at the expense of an increased chance of loss and more variable queueing delay. It should be noted that, although the virtual queue scheme is simple, it is robust at detecting congestion under a wide variety of traffic types. It also has an operating range which is not extremely sensitive to the setting of the parameters B and θ.

4. PROBING IMPLEMENTATION

Probing achieves several aims: to get a sample of the current level of congestion on the route and to infer if the network links along the route can actually carry the peak rate of the traffic. Probing also allows a host to determine if end-to-end latency requirements can be met[5]. It should be considered as part of the wider connection setup protocol that applications use; for example the data involved in a Session Initiation Protocol exchange could be carried in the probe packet payloads.

Suppose that host A wishes to send a real-time stream to host B with a peak rate R and packet size s. This will involve sending a series of probe packets from A to B in an attempt to infer the level of congestion. Host A will initiate the call if the proportion of probe packets that experienced congestion is less than a threshold level, ϵ. Probe packets are UDP packets with the ECN capable bit set in the IP header, a sequence number, and contain a timestamp for calculating the RTT (round trip time). These UDP packets are padded to be of size s. Host B acts as a reflector for host A's probe packets returning timestamp information to calculate the RTT and echoing any congestion notifications received by the probe packets. (Notice this extends to bi-directional real-time streams where the reflected packets could be merged with probe packets for the return stream from host B.)

Host A initiates the probing by sending a packet to host B requesting to open a probe stream[6]. Throughout the probe, host A calculates the current proportion of acknowledged packets that experienced congestion, p_{total} and a RTT estimator. Let n be the total number of probe packets acknowledged. When host A updates p_{total}, if

$$p_{total} > \epsilon + \frac{1}{n} \tag{1}$$

then host A ceases to send probe packets and blocks the call. Furthermore, should host A detect a dropped packet through a break in acknowledged sequence numbers it blocks the call[7]. Let both these types of blocking be called "early-abort". Early-abort is designed to prevent the probe traffic from adding to congestion. The importance of mechanisms to avoid probes causing additional congestion is explored in [11].

[5]Providing queuing delays are bounded by appropriate setting of buffer sizes (B as in Section 3) throughout the network.

[6]This initial packet is sent with a timeout value so as to be resilient to packet loss. Upon loss of the first packet the connection is blocked.

[7]The use of sequence numbers to detect loss is not resilient to packet reordering and a timeout based mechanism (such as that used in TCP) would be more appropriate if it were a problem. Further strategies involving the calculation of drop probability could be considered, which is important for channels with high bit error rates, but the aim of this study is to show that a congestion controlled IP network can support multiple traffic types with low delay and low loss.

Upon acknowledgement of the first probe packet, the probing is then broken up into rounds which last approximately one RTT each. The transition from one round to the next occurs when an acknowledgement for a probe packet in the current round is received by host A. During each round, host A calculates the proportion of packets sent in the last round which experienced congestion, p_{last}. At the transition, host A increments the rate at which it sends to $(2 - p_{last}) * r$, where r is the rate at which host A was sending in the previous round. For the initial round host A is assumed to be sending at approximately rate s/RTT. The rate thus increases exponentially and is dependent on the level of network congestion. The higher the level of congestion the more cautious the rate increase. This is aimed at limiting probe induced congestion.

When making round transitions, if host A's sending rate equals or exceeds the target rate R then host A sends at rate R and notes that this is the penultimate round[8]. Having probed for a round at rate R, upon the next round transition host A enters a final round. The probing terminates when A has entered the final round and has received acknowledgements for at least $\max(\frac{1}{\epsilon}, k)$ in that final round. The $\frac{1}{\epsilon}$ lower bound is used to improve the fairness between connections with disparate RTTs using a congested resource. This is because the blocking probability of a connection is affected by the number of probe packets sent (this is examined further in Appendix B). The lower bound also helps routes with short RTT to infer the route's capacity by exercising the various links.

Upon termination, if the final marking proportion p_{total} does not exceed ϵ and the probe has not aborted early, then host A initiates its data stream. Otherwise the call is blocked.

The time taken to achieve a decision with this probing algorithm is dependent on the packet size s, the target rate R, the level of congestion expressed as a marking probability p, and the round trip time RTT. On average the scheme will achieve an acceptance decision in time of order

$$RTT * \left(\left\lceil \log_{2-p} \left(\frac{R * RTT}{s} \right) \right\rceil + 2 \right) \text{seconds.} \tag{2}$$

The dependence on round trip time of the connection setup time is natural as a route with a longer round trip time must wait longer to receive probe feedback. For a target rate of 80Kbps, packet size of 200 bytes, round trip time of 50ms and a marking probability of 0.1, an acceptance decision takes around 0.2 seconds. This compares favorably with the probing schemes put forward in [6, 11, 12] which suggest probe times that are static and larger. These other schemes have placed emphasis on reducing the size of the confidence interval for the marking probability estimator and hence the derived congestion level. This probe protocol is designed to make connection setup quicker while the class of acceptance controlled traffic maintains good behaviour in aggregate. By making the probe time shorter, probing becomes more attractive as a deployable connection acceptance method.

It should be noted that probe packets need not be wasting transmission capacity. Other call setup protocols could be carried in the payload of the probe packets themselves, although this may increase the complexity of the probing implementation. Despite the low number of probe packets it would be reasonable to transfer about 2KB of setup information in each direction assuming a

[8]Note that the acknowledgements for packets sent during this penultimate round will only be received during the final round.

RTT of 30ms, a rate of 80Kbps and packets of total size 200 bytes (assuming around 10 acknowledged probe packets).

The setting of the parameter ϵ is dependent on policy. A larger ϵ corresponds to a less cautious attitude towards congestion and hence a lower blocking probability for a given marking probability. Appendix B contains a mathematical analysis of the effects of ϵ on blocking probability. Whether ϵ should be set centrally or by each individual end-system is an open issue concerning the control of network resource allocation.

Both probe protocols described in [6, 12] require the provisioning of a DiffServ enabled network with suitable scheduling mechanisms. The probing is then performed at a lower priority level than data. In contrast the probing presented here is not premised on DiffServ deployment, merely on ECN mechanisms in routers, but will also function in a DiffServ environment. Furthermore the probe times of the protocol presented here are self-scaling to the round trip time of the requested route.

The protocol designed here builds on the in-band probing results presented in [11]. In particular congestion is more actively detected throughout the probing. The protocol is also more cautious in its rate increases when it senses possibly high congestion levels. Furthermore the protocol is designed to set up connections quickly and makes few assumptions about network topology.

5. SIMULATION METHOD

The purpose of the simulations was to show the viability of distributed connection acceptance control in both an IP network partitioned to support a single traffic type (e.g. a DiffServ class or a voice over IP telephone network) and in a mixed traffic integrated service IP network. The results should be viewed qualitatively and not quantitatively. They are intended to show that queueing delay and loss rates for interactive applications can remain controlled in both partitioned and integrated networks using ECN mechanisms and the lightweight protocol described. It is not the aim of this paper to determine optimal values for θ, B, and ϵ which will inherently depend on network policy and traffic patterns.

The real-time call traffic, representing voice telephony, was simulated with two types of source model. The first was a CBR (constant bit rate) stream where 200 byte packets were sent at a rate of 80Kbps. This corresponds to a voice channel sampled at 8KHz with 8 bits per sample where each packet is a 20ms frame with a header overhead of 40 bytes. The second type of source modeled voice traffic with silence suppression. In this model a source is either "on" or "off", representing a talk-spurt or silence respectively. During the "on" periods the source would send 200 byte packets at a rate of 80Kbps as in the CBR model, while in the "off" periods no data was sent. The length of both the "on" and "off" periods were independent exponential random variables with mean 500ms.[9]

Real-time calls (with a source and destination host) arrived according to a Poisson process with intensity λ. On arrival the source host would conduct a probe of the network route to the destination host using the protocol presented in Section 4. The probe would then either abort or complete. If the probe completed, a call (with one of the source properties described above) would start and last for a call holding time which was exponentially distributed with mean 200 seconds. If the probe aborted the call would be blocked and not enter the network.[10]

The simulations were run at various average offered load levels expressed as a fraction of raw link rate. Consider the load created by the CBR call traffic at 80Kbps which had mean call holding time τ. This has an arrival process which is Poisson with intensity λ. Hence the average load induced by this class of traffic is $\tau.\lambda.80$Kbps. Similarly the exponential "on"/"off" traffic created an average load of $\tau.\lambda.40$Kbps.

For the integrated service simulations TCP flows conforming to the RFC2481 ECN standard [16] were used[11]. This scheme for TCP's reaction to ECN marks was chosen as it is implemented and starting to be deployed in end systems today, see [13]. The TCP flows used Reno congestion control with ECN extensions, had a receive window of 20 packets with segments of size 1000 bytes.

Two models of TCP connection arrival were simulated. In the first, TCP connections arrived according to a Poisson process with intensity μ and proceeded to transfer 1MB of data. This produced an offered load of $\mu.8$ Mbps. This arrival model is non-elastic in nature[12] and would correspond to networks being used for machine driven bulk transfers such as scheduled backups or mirroring.

The second model attempted to generate TCP connections as would be generated by active web users. The web user model was motivated by the SURGE web workload generator [5] and studies of web traffic characteristics in [4]. In this model an active web user downloads a page which consists of the page itself and a number of embedded pages using a persistent connection. Such a download corresponded to a single TCP connection which transfered $\sum_{i=1}^{X+1} Y_i$ bytes, where X is the number of embedded files in a page and Y_i is the distribution of transferred file sizes in bytes. The user then reads the page for a random amount of time before downloading the next page and so on. A table displaying the distributions and parameters used to generate traffic for a web user is displayed in Table 1.

The topologies used for simulations are shown in Figures 1 and 2. For all simulations ϵ was set to 0.1, B to 32KB, and C to 30Mbps for the bottleneck link. The back-path on the bottleneck link also had capacity 30Mbps and identical delay, but did not have any active queue management. All other links did not have any active queue management and were symmetric in link speed and delay.

All simulations were run for 4800 seconds with the first 1200 seconds of data discarded while the system stabilised.

[9] In [19] it is concluded that using an exponential "on"/"off" model is broadly valid for qualitative voice over IP analysis but that for quantitative results the talk-spurts and silences require the use of "heavy-tailed" distributions with parameters dependent on the voice codec used.

[10] This is a simplified model and actual studies of telephone traffic, such as [8], suggest that a lognormal call holding time with mean between 200-300 seconds is more appropriate. Simulations run with such a "heavy-tailed" distribution showed negligible difference when compared with all the results shown for the simpler exponential model.

[11] The ns-2 FullTCP class was used as it best approximates TCP implementations and includes the SYN and FIN interchanges.

[12] Notice that each TCP connection is elastic in nature but the demand for data transfer by the class of TCP traffic is on average constant at $\mu.8$ Mbps

Component	Distribution	Probability density $p(x)$	Parameters	Mean
Interarrival times (sec)	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}, x > k$	$k = 1.0, \alpha = 1.5$	3.0
Embedded references	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}, x > k$	$k = 1.0, \alpha = 2.43$	1.70
Request file sizes (bytes)	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}, x > k$	$k = 800, \alpha = 1.11$	8070

Table 1: Summary of distributions and parameters used in the web user TCP connection model.

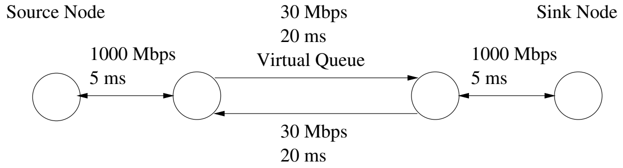

Figure 1: Topology for single round trip time simulations.

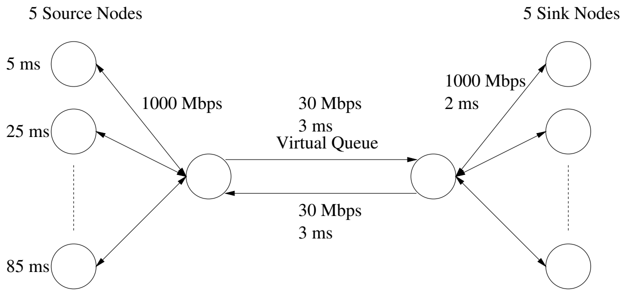

Figure 2: Topology for multiple round trip time simulations.

6. RESULTS

Three scenarios were considered to investigate the performance of the admission control protocol within an IP network (or DiffServ class) dedicated to voice traffic. Two simulations were performed for a single bottleneck link. In the first case all source-sink pairs shared the same round trip time (see Figure 1), while in the second various round trip times were considered (see Figure 2).

6.1 Bottleneck link in a dedicated network

Each single bottleneck scenario was simulated with both CBR and exponential "on"/"off" traffic sources to show the applicability of probing to traffic with differing source characteristics.

In the single bottleneck with a single round trip (as in Figure 1) the simulations were run for $\theta \in \{0.75, 0.80, 0.85, 0.90, 0.95\}$ and load levels from $0.5C$ to $1.5C$ in $0.1C$ increments, where C is the capacity of the bottleneck. To vary load levels the intensity of the Poisson arrival process, λ, was set accordingly (see Section 5).

Plots of blocking probability to offered load for both CBR and exponential "on"/"off" traffic are shown in Figures 3 and 4 respectively. Notice the relationship between setting the scaling factor θ of the virtual queue and blocking probability; as θ increases the blocking probability decreases for a given load level. The number of call seconds carried is shown in Figures 5 and 6. As expected the number of calls carried in the exponential "on"/"off" case is about twice that for the CBR calls at a given load level. Comparing the blocking probability and call seconds carried for CBR traffic with that for exponential "on"/"off" traffic leads us to conclude that the probing is acceptable for either characterisation as accepted load is controlled in both cases. This suggests robustness of the scheme to a variety of traffic characterisations.

The average queuing delay experienced for the CBR and exponential "on"/"off" sources is shown in Figures 7 and 8 respectively. It can be seen that queueing delay is controllable by setting the virtual queue parameter θ[13]. Varying θ displays a policy tradeoff between carrying more calls (shown by a lower blocking probability for a given load) and increased average queueing delay. The maximum loss rate observed was 0.005% for a load level of $1.5C$ and θ of 0.95 with exponential "on"/"off" traffic sources. Hence in the partitioned network model, the scheme maintains a low loss rate and low average delay as required.

When analysing any signalling scheme it is important to consider how much traffic the signalling protocol itself generates. The percentage of total traffic which is probe traffic, for a mean call holding time of 200 seconds, is 0.21% and 0.43% for CBR and exponential "on"/"off" sources respectively at a load level of $1.5C$ and a θ of 0.8. This represents an acceptable overhead even when heavily loaded. It would be possible to use the probe packets for endpoint call setup negotiation. However the scheme's overhead would be unacceptable for flows in which the mean call data transferred is short.

To consider the effects of round trip time on the probing scheme, simulations (as in Figure 2) were run for a virtual queue θ value of 0.8 and load levels from $0.5C$ to $1.5C$ in $0.1C$ increments. Load was assumed to come from each of the five source nodes equally. Hence a Poisson arrival process with intensity $\frac{\lambda}{5}$ was used at each source node to generate a Poisson arrival process with intensity λ at the bottleneck.

Plots of blocking probability for the CBR and exponential "on"/"off" traffic are shown in Figures 9 and 10 respectively. It can be seen that connections with shorter round trip times generally have a lower blocking probability than connections with longer round trip times for the same load level. This can be explained by the positive relationship between the number of probe packets sent and the round trip time. As analysed in Appendix B, the blocking probability for a given marking probability is affected by the number of probe packets sent. It should be noted that the unfairness is minor in comparison to the rate bias TCP shows towards connections with short round trip times. To make the scheme fairer between connections with short and long round trip times more probe packets could be sent, but this would increase the connection setup time of connections with shorter round trip times. The increasing of probe traffic and connection setup time should be set against the fact that unfairness is negligible unless the system is subjected to overload.

[13] Setting the buffer size, B, is also a factor. Here we have assumed that it is set to 32KB which gives about 9ms of queueing delay in the worst case.

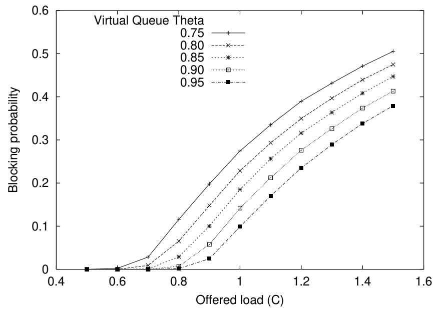

Figure 3: Blocking probability for simple bottleneck link and CBR sources.

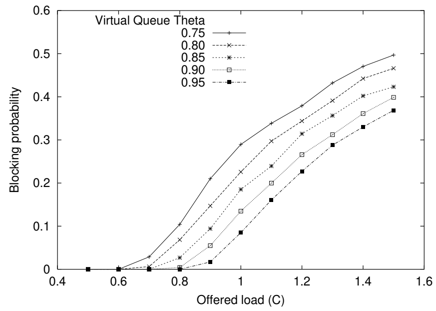

Figure 4: Blocking probability for simple bottleneck link and exponential “on”/“off” sources.

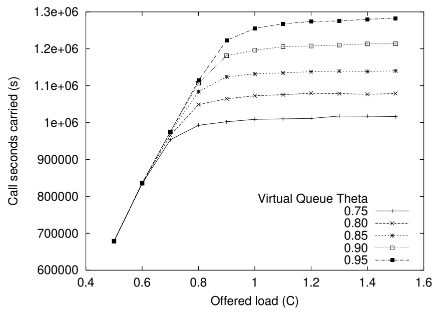

Figure 5: Call seconds carried for simple bottleneck link and CBR sources.

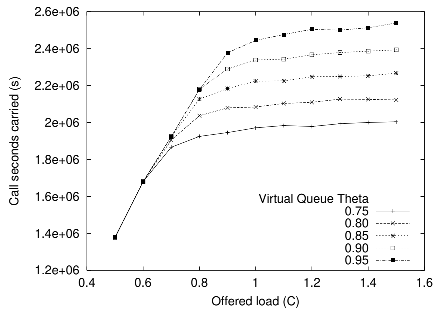

Figure 6: Call seconds carried for simple bottleneck link and exponential “on”/“off” sources.

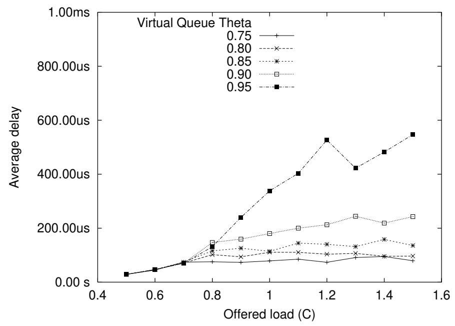

Figure 7: Average delay for simple bottleneck link and CBR sources.

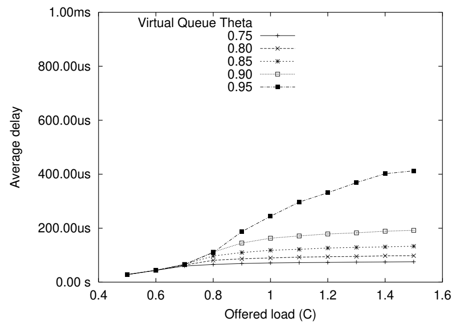

Figure 8: Average delay for simple bottleneck link and exponential “on”/“off” sources.

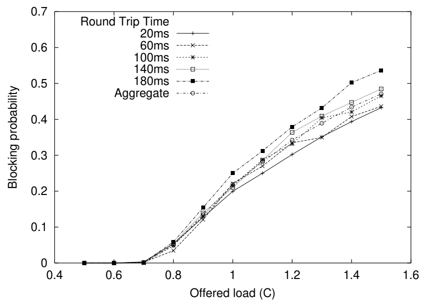

Figure 9: Blocking probability for single bottleneck link and CBR sources with multiple round trip times.

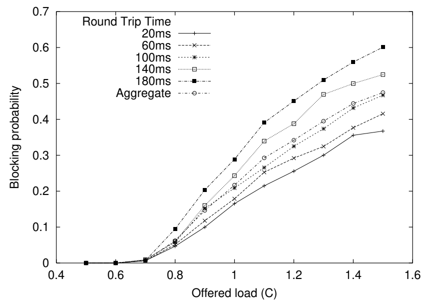

Figure 10: Blocking probability for single bottleneck link and exponential “on”/“off” sources with multiple round trip times.

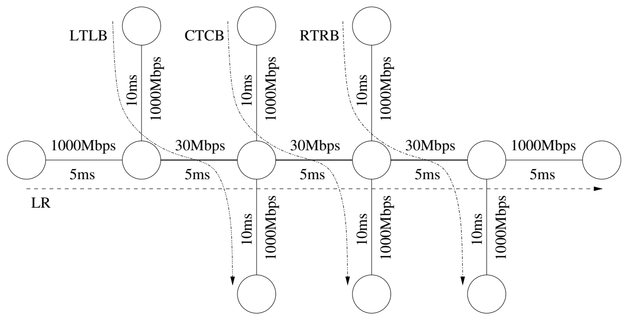

Figure 11: Topology and flow routes for multi-hop bottleneck simulations.

6.2 Multi-hop bottlenecks

To consider the effects on the admission control protocol for flows passing through multiple congested links, simulations based on the topology shown in Figure 11 were used. Each bottleneck link running at 30Mbps had a virtual queue in the forward direction with B set to 32KB and θ set to 0.8. Connections arrived to be routed either along the multi-hop path or across the multi-hop path with equal load. As before load was set by altering the intensity of each individual Poisson arrival process.

It is to be expected that the multi-hop connections will have a higher blocking probability than the single hop connections because they are passing through multiple congested resources while the single hop connections pass through only one.[14] The blocking probabilities for connections arriving for each path are shown in Table 2. The multi-hop connections have a higher blocking probability than the single hop connections but this discrimination is not as severe as that seen in per-hop measurement-based acceptance control schemes [10]. In per-hop measurement-based acceptance control schemes the acceptance probability for the multi-hop flow will be roughly the product of the acceptance probabilities at each hop. To compare this with the end-to-end probing scheme presented here the product form is shown in the last column of Table 2. The product form in some cases overestimates the blocking probability by around 20%. This suggests that the probe-based scheme does not discriminate against multi-hop flows by as much as per-hop measurement-based schemes. Whether multi-hop connections should be blocked more readily on traversing multiple congested links is a potentially interesting policy issue not addressed here.

6.3 Bottleneck link in an integrated network

Two scenarios were used to investigate the performance of endpoint probing in a network also carrying TCP traffic. Both scenarios used the topology displayed in Figure 2. In both scenarios, interactive call traffic, as described in Section 5, arrived according to a Poisson process of intensity $\frac{\lambda}{5}$ at each source node. This provided a call arrival rate of λ at the bottleneck link with connections having a range of round trip times. The simulations were run with both CBR and exponential "on"/"off" traffic sources. The two scenarios differed in the TCP arrival model being used. In the first scenario TCP connections wishing to transfer 1Mb of data arrived according to a Poisson process of intensity μ. While in the second scenario TCP traffic was generated by N active web users using the model of web sessions as described in Section 5.

In the Poisson TCP arrival simulation, load was set as a fraction of capacity by varying λ and μ appropriately. The TCP traffic load was set to be 50% of the arriving load. As the TCP connections were not acceptance controlled the TCP load accounted for more of the work done by the network when contention for resources occurred. The simulations were run for aggregate load levels of $0.4C$ to $1.2C$ in $0.2C$ increments. The virtual queue at the bottleneck had θ set to 0.9.

The results for this simulation are displayed in Table 3. As desired the queueing delay remains controlled and the drop rate low even when the offered load exceeds the link capacity. This means that accepted call traffic, regardless of the network load, will receive an adequate network latency and loss rate. When the system is moderately loaded ($0.4C$ to $0.6C$) the call traffic and TCP traffic receive an approximately equal share of the network throughput and call blocking probability remains reasonable. However as the load increases the TCP traffic takes an increasing share of the network throughput and the call traffic is effectively shut out of the system. This is because the TCP traffic is not acceptance controlled and so all work arriving for TCP transference will be carried regardless of network load[15].

For the simulation displaying the integration of web and call traffic, load could not be set as simply. The problem arises because the connection arrivals of web traffic are elastic with respect to network load; a user can only download the next page once the current page is downloaded. Hence the rate at which TCP connections are offered to the network depend on the current loading of the network. To get an idea of scale, a link of size 30Mbps can handle 600 concurrent 50Kbps streams. The simulations were run for pair loadings $(0.2C, 250)$ to $(0.6C, 1250)$ in increments of $(0.1C, 250)$, where the pair (xC, n) corresponds to a interactive call load of xC and n web users[16]. The virtual queue at the bottleneck had θ set to 0.9.

The results for the system carrying call and web traffic are displayed in Table 4. Again the average queueing delay and drop rate remains controlled in all circumstances. This means that call traffic which is accepted will receive satisfactory end-to-end latency and loss rates. Call and web traffic receive an approximately equal share of the network throughput and call blocking probability remains reasonable while the system is lightly loaded (($0.2C$, 250) to ($0.3C$, 500)). As the network is loaded from ($0.4C$, 750) to ($0.6C$, 1250) call traffic is increasingly blocked out of the network, because the web traffic is not acceptance controlled. The web traffic still arguably receives some increased aggregate utility as the total number of flows completed rises[17]. At the high loading level

[14] Assuming that the probability of marking at each queue is independent of other queues. The probability that a packet sent across the multi-hop connection is unmarked is the product of the probabilities that the packet is unmarked across each one hop link. Thus the probability that the multi-hop probe packets are marked will be greater and this leads to a higher blocking probability for the multi-hop connections.

[15] Whether TCP connections should also be acceptance controlled is an interesting policy issue with wide implications. One consistent framework for considering network utility with competing traffic types is through network pricing. Congestion pricing schemes for networks with congestion notification are discussed in [18, 21, 22].

[16] The number of web users was chosen by running simulations with the web users in isolation and determining the fraction of capacity they used so that it roughly equaled xC.

[17] It could be argued that web utility is not correctly modeled by TCP's current response and there is a real time threshold at which

Load	Flow LTLB	Flow CTCB	Flow RTRB	Flow LR	Product
$(0.3C, 0.3C, 0.3C, 0.3C)$	0.0000	0.0000	0.0000	0.0000	0.0000
$(0.4C, 0.4C, 0.4C, 0.4C)$	0.0384	0.0274	0.0412	0.0963	0.1033
$(0.5C, 0.5C, 0.5C, 0.5C)$	0.1390	0.1218	0.1663	0.3172	0.3696
$(0.6C, 0.6C, 0.6C, 0.6C)$	0.2198	0.2449	0.2284	0.4865	0.5454

Table 2: Blocking probabilities for multi-hop (Flow LR) and single bottleneck flows (Flows LTLB, CTCB, and RTRB) sharing resources with an exponential "on"/"off" source model. The last column corresponds to a blocking probability calculated as the product of the individual acceptance probabilities seen by the single bottleneck flows.

Load	Utilisation (%)	Drop probability	Mean queueing delay	Call blocking probability	Call seconds carried	TCP traffic (% of total)	TCP flows completed
$0.4C$	41.99	0.0000	0.2090ms	0.0040	550235	50.79	2768
$0.6C$	59.69	0.0000	0.4031ms	0.0605	763908	52.45	4065
$0.8C$	70.58	0.0002	0.7531ms	0.2838	766560	59.91	5488
$1.0C$	71.76	0.0019	1.1735ms	0.6268	503182	73.62	6834
$1.2C$	72.94	0.0061	1.5166ms	0.8476	261976	86.56	8172

Table 3: TCP traffic with Poisson arrival process and exponential "on"/"off" call traffic.

of $(0.6C,\ 1250)$, only 36% of the offered calls are being carried while all web traffic is carried. The web users do see some slight degradation in service as the number of flows completing does not increase as dramatically.

The results for integrated service networks presented here suggest that queueing delay can be controlled in single class integrated networks. This is at the expense of reduced link utilisation. The performance of the integrated network is good for moderate loading with acceptable call blocking and good TCP performance. However as the load rises the TCP traffic takes an increasing share of the traffic, making the call blocking probability prohibitively high. This illustrates part of a wider Internet issue involving competing traffic types contending for network resources. It suggests that given sensible dimensioning (and hence knowledge of network load patterns) the running of a low loss and low delay IP network is viable.

7. LIMITATIONS AND FURTHER WORK

The scheme for a possible evolution towards integrated services presented here is not without limitations. Policing the network to ensure that untrusted end-systems are adopting appropriate congestion and acceptance control is difficult. But this is an open research issue for the current Internet as well [15]. Congestion pricing is a framework for ensuring that end-systems have an incentive to regulate themselves appropriately, see [18, 22]. Within a congestion pricing regime the probe protocol presented here is estimating the spot price of a call and the threshold ϵ represents the amount the end-system is willing to pay.

Explicit centralised control of network policy[18] is an open issue in the environment described. By leaving all connection acceptance and congestion control to end-systems the centralised setting of network policy becomes more difficult. It should however be possible to construct such a policy framework on top of ECN based protocols by either setting ϵ appropriately for non-elastic traffic (as mentioned in Appendix B) or using differentiated congestion control responses in TCP (as put forward in [21, 22]) to achieve differing bandwidth shares. In a dedicated network environment, where the network manager controls the network and the end system, this would involve the distribution of appropriate parameters to the end-systems. In an open network the issue becomes more thorny: whether such differing responses should be controlled centrally or according to end-system utility is an open issue.

The probe connection acceptance control described here does not provide connections with a hard guarantee of service or an exact measure of network conditions. The use of the probe allows connections to infer whether it is sensible to enter the network and this service model may be unacceptable for certain applications (e.g. remote surgery).

There remain several limitations involving "system size" and timescales. In this context system size refers to the relative size of each non-adaptive flow compared to the route capacity. If the system size is small (i.e. only a few non-adaptive flows will saturate the link) then the potential cost of a mistaken acceptance decision could be large. The timescale on which flow arrival and departure events occur from the class of all non-adaptive flows affects the rapidity with which this class can adapt its load on the network[19]. Increased levels of aggregation improves both system size effects and the rapidity of adaptation that the class of non-adaptive flows has to link conditions. In certain settings system size and timescale effects may degrade the performance of the congestion control mechanism to act as an effective adaption mechanism for the class of non-adaptive traffic.

From the results presented here, ECN based integrated service networks appear viable and attractive in terms of their ability to act as an evolution of the Internet towards a multi-service network. The use of other marking algorithms is appealing and should be studied. Adaptive marking algorithms such as those suggested in [3, 24] may enable higher utilisations to be achieved while maintaining low loss and low delay within an integrated network, while the use of different ECN responses for TCP might also lead to per-

users cease a download.

[18] In this context network policy refers to decisions such as "I would like my TCP traffic to take at most 50% of link capacity during times of contention".

[19] For example, decreasing the mean call holding time while keeping the total load constant will increase the number of arrival and departure event within the class of non-adaptive traffic making the class more rapidly adaptive to network conditions.

Load	Utilisation (%)	Drop probability	Mean queueing delay	Call blocking probability	Call seconds carried	TCP traffic (% of total)	TCP flows completed
$(0.2C, 250)$	35.34	0.0000	0.1718ms	0.0004	522577	43.58	261392
$(0.3C, 500)$	62.76	0.0004	0.6086ms	0.0635	760929	54.88	503738
$(0.4C, 750)$	71.91	0.0018	1.0282ms	0.2796	783943	59.05	673029
$(0.5C, 1000)$	73.60	0.0035	1.2546ms	0.4798	713702	63.69	785330
$(0.6C, 1250)$	73.68	0.0060	1.4523ms	0.6304	592925	69.60	865345

Table 4: TCP traffic generated with web user model and exponential "on"/"off" call traffic.

formance enhancements. The use of the scheme in environments with low aggregation needs to be studied. The mapping of policy settings into real paramters for network control is also worth exploring. Finally issues with incremental deployment remain, but the outlook is promising.

8. CONCLUSION

The protocol presented is based on in-band probing of an ECN enabled IP network to infer network topology and congestion levels. Care is taken to ensure that probing does not in itself lead to excess congestion. The connection setup speed aims to be fast and is dependent on the underlying round trip time. The policy on actual acceptance thresholds is expressed through a single parameter, ϵ, and can fit into both a centrally administered network or a decentralised network using a congestion pricing framework [18, 21, 22].

The simulation analysis suggests that running a partitioned IP network for interactive traffic using the distributed connection acceptance control protocol would be possible. The use of the protocol in a more general integrated network has been studied and it appears viable to run a low delay and low loss IP network using ECN mechanisms. The scheme has scaling, fault tolerance and flexibility advantages over traditional integrated service schemes.

The approach is not without its limitations (see Section 7) but many of these also affect the current Internet. However the scheme provides an evolution of current Internet protocols towards an integrated service environment while maintaining the heterogeneity, scalability, and fault tolerance properties which were central to the original design and success of the Internet.

9. ACKNOWLEDGEMENTS

This work was done while a Royal Commission for the Exhibition of 1851 Industrial Fellow at AT&T Research Labs, Cambridge. Thanks go to Andy Hopper, Frank Kelly, and Glenford Mapp for many useful discussions and Richard Mortier for his proof reading and advice on ns simulations. Thanks also go to the area editor and the anonymous reviewers for their constructive comments.

10. REFERENCES

[1] NS (Network Simulator). Available via `http://www.isi.edu/nsnam/ns/`.

[2] U. Ahmed and J. Hadi Salim. Performance evaluation of explicit congestion notification (ECN) in IP networks. *Internet RFC 2884*, July 2000.

[3] S. Athuraliya and S. H. Low. Optimization flow control, II: Implementation. Submitted for publication. Available via `http://netlab.caltech.edu/netlab-pub/rem2.ps.gz`, May 2000.

[4] P. Barford, A. Bestavros, A. Bradley, and M. Crovella. Changes in Web client access patterns: Characteristics and caching implications. *Special Issue on World Wide Web Characterization and Performance Evaluation; World Wide Web Journal*, December 1998.

[5] P. Barford and M. Crovella. Generating representative Web workloads for network and server performance evaluation. In *ACM SIGMETRICS 1998*, pages 151–160, Madison, WI, July 1998.

[6] G. Bianchi, A. Capone, and C. Petrioli. Throughput analysis of end-to-end measurement-based admission control in IP. In *IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[7] S. Blake, D. Black, M. Carlson, E. Davis, Z. Wang, and W. Weiss. An architecture for differentiated services. *Internet RFC 2475*, December 1998.

[8] V. Bolotin. Telephone circuit holding time distributions. In *Preceedings of the 14th International Teletraffic Congress*, pages 125–134. Elsevier, June 1994.

[9] R. Braden, D. Clark, and S. Shenker. Integrated services in the Internet architecture: An overview. *Internet RFC 1633*, July 1994.

[10] L. Breslau, S. Jamin, and S. Shenker. Comments on the performance of measurement-based admission control algorithms. In *IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[11] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang. Endpoint admission control: Architectural issues and performance. In *ACM SIGCOMM 2000*, pages 57–69, Stockholm, Sweeden, October 2000.

[12] V. Elek, G. Karlsson, and R. Ronngren. Admission control based on end-to-end measurements. In *IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[13] S. Floyd. ECN (explicit congestion notification) in TCP/IP. Web page `http://www.aciri.org/floyd/ecn.html`.

[14] S. Floyd. TCP and explicit congestion notification. *ACM Computer Communication Review*, 24(5):10–23, October 1994.

[15] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the internet. *IEEE/ACM Transcations on Networking*, 7(4):458–472, May 1999.

[16] S. Floyd and K. Ramakrishnan. A proposal to add explicit congestion notification (ECN) to IP. *Internet RFC 2481*, January 1999.

[17] R. J. Gibbens and F. P. Kelly. Distributed connection acceptance control for a connectionless network. In *Proceedings of the 16th International Teletraffic Congress*, pages 941–952. Elsevier, June 1999.

[18] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35:1969–1985, 1999.

[19] W. Jiang and H. Schulzrinne. Analysis of on-off patterns in VoIP and their effect on voice traffic aggregation. In *The 9th IEEE International Conference on Computer Communication Networks*, 2000.

[20] F. Kelly, P. Key, and S. Zachary. Distributed admission control. *IEEE Journal on Selected Areas in Communications*, 18(12):2617–2628, December 2000.

[21] F. P. Kelly. Models for a self-managed internet. In *Philosophical Transactions of The Royal Society A358*, pages 2335–2348. The Royal Society, August 2000. Available via `http://www.statslab.cam.ac.uk/~frank/smi.html`.

[22] P. Key and D. McAuley. Differential QoS and pricing in networks: where flow-control meets game theory. *IEE Proceedings Software*, 146(2), March 1999.

[23] S. Kunniyur and R. Srikant. End-to-end congestion control schemes: Utility functions, random losses and ECN marks. In *IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[24] S. Kunniyur and R. Srikant. A time-scale decomposition approach to adaptive ECN marking. In *IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[25] T. Ott. ECN protocols and the TCP paradigm. In *Workshop on Modeling of Flow and Congestion Control Mechanisms*. Ecole Normale Supérieure, September 2000.

[26] D. Wischik. How to mark fairly. In *Workshop on Internet Service Quality Economics*. MIT, September 1999. Available via `http://www.statslab.cam.ac.uk/~djw1005/Stats/Research/marking.ps`.

APPENDIX

A. A COMPARISON OF LOSS AND DELAY FOR ECN ENABLED AND TRADITIONAL IP NETWORKS

The following simulations try to motivate the loss and delay advantages of running an IP network with ECN support.

The simulations used the network topology shown in Figure 2. Each of the five source-sink pairs had N TCP connections attached. In the ECN-aware case, RFC2481 compliant ECN-TCP connections were used and a virtual queue with θ=0.9 and B=32KB. In the traditional case, the TCP connections used Reno congestion control and the bottleneck had FIFO queueing with a buffer size of 32KB. The simulations were run for 180 seconds with the first 60 seconds of data discarded while the system stabilised.

Cumulative delay distributions[20] for all packets arriving at the queue are shown in Figures 12 and 13. Notice that the cumulative delay distributions for the ECN-aware case are predominantly concave, whereas for the traditional case the cumulative delay distributions are predominantly convex. This has a direct influence on the number of late packets that a real-time application will experience[21]. For example, with 64 TCP connections the number of packets experiencing over 6ms of delay is about 10% for the ECN case and around 50% for the traditional case. Although the above analysis could be considered simplistic, it is the ability to change the general shape of the delay distribution curve that matters. This motivates researchers to study the possibility of low loss and low delay single class networks with explicit congestion notification.

These results are in line with those presented in [14] which has results on the throughput and delay performance of ECN using Random Early Detection queues. The interested reader can find ECN performance results based on an ECN-TCP implementation in [2].

B. PACKETS ACKNOWLEDGED AND BLOCKING PROBABILITY

The blocking probability of the acceptance control presented in Section 4 has a dependence on the number of probe packets acknowledged (and so sent). Suppose that, for a given probe session, probe packets are marked with probability p. This assumption is valid providing the queueing timescales are smaller than the time between probe packets and that coarse load varies on timescales larger than probe session times. The probability of an early abort at or before the arrival of the k^{th} acknowledgement is

$$1-\sum_{i=0}^{k}(1-p)^{k-i}\,p^{i}A\,(i,k) \tag{3}$$

where $A(i,k)$ is the number of sequences of k packets such that if there are i marks the probe does not early abort. Assuming $0<\epsilon<1$, $A(i,k)$ is given by

$$\begin{aligned} A(0,k) &= 1\\ A(1,k) &= k\\ A(i,k) &= \begin{cases} 0 & \text{if } i>k\epsilon+1\\ A(i-1,k-1) & \text{if } i=k\epsilon+1\\ A(i-1,k-1)+A(i,k-1) & \text{if } i<k\epsilon+1 \end{cases} \end{aligned}$$

The final decision on acceptance is taken when the probing terminates and compares the proportion of marked packets, p_{total}, with the threshold ϵ. Extending from the probability of early abort (3) to blocking probability requires the exclusion of the terms corresponding to a proportion of marked packets exceeding ϵ. If the decision is taken after K probes the blocking probability is thus

$$1-\sum_{i=0}^{\lfloor\epsilon K\rfloor}(1-p)^{k-i}\,p^{i}A\,(i,K) \tag{4}$$

The number of packets used in the probe against the blocking probability experienced for ϵ values of 0.05, 0.1, and 0.25 are shown in Figures 14, 15, and 16 respectively.

It is a design objective that the connection setup time be made as short as possible. So sending as few packets as possible is desirable. However the speed at which probes hit the target rate is dependent

[20] Note that the cumulative distribution plots shown include packet drops. Such dropped packets are considered as packets with infinite delay.

[21] Here late packets are those that either never arrive at their destination or arrive too late to be pertinent to the user application.

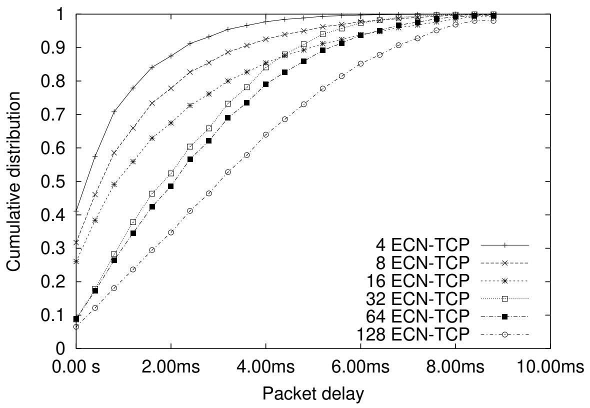

Figure 12: Cumulative delay distribution of packets arriving at bottleneck link for various numbers of TCP connections at each source in an IP network using ECN mechanisms.

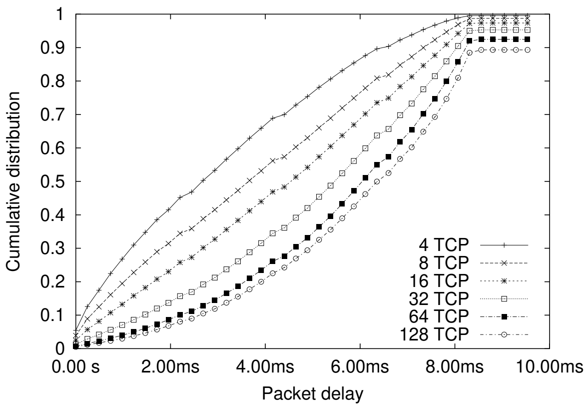

Figure 13: Cumulative delay distribution of packets arriving at bottleneck link for various numbers of TCP connections at each source in an IP network using traditional congestion control.

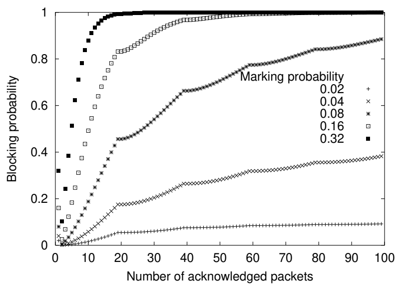

Figure 14: Blocking probabilities for $\epsilon = 0.05$.

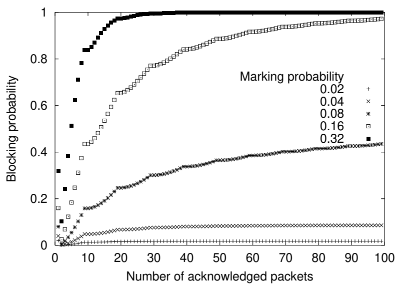

Figure 15: Blocking probabilities for $\epsilon = 0.1$.

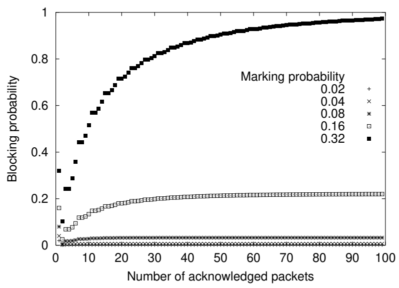

Figure 16: Blocking probabilities for $\epsilon = 0.25$.

on the round trip time of the connection. There is thus a tradeoff between the minimum acceptable connection setup time and ensuring blocking probability fairness between connections with short and long round trip times. Notice that the probes which make a decision based on less than $\frac{1}{\epsilon}$ packets have a significantly lower blocking probability compared to those which use more than $\frac{1}{\epsilon}$ packets. The design decision to probe for at least $\frac{1}{\epsilon}$ packets in the last round helps to balance the lower blocking probabilities for connections with a short round trip time against the desire for short setup times. Although not providing complete fairness across all marking probabilities and all values of ϵ, it performs well when the congestion levels are not approaching overload.

From Figures 14, 15, 16 it can be seen that the blocking probability of a connection is dependent on its value of ϵ. This allows a differentiated service model to be built by having differing ϵ values at end-systems. For example suppose there are two classes of connections, one using ϵ_1 and the other using ϵ_2 where $\epsilon_1 < \epsilon_2$. For a given marking rate, the blocking probability of connections using ϵ_2 will be lower than those for ϵ_1. This method of differentiated service has assumed that the end-systems are trusted to use values of ϵ which don't lead to the system stabilising at dangerously high congestion levels. In a non trusted environment congestion pricing offers an appropriate incentive for end-systems to choose a suitable value of ϵ.