Network Working Group
Request For Comments: 2211
Category: Standards Track

J. Wroclawski
MIT LCS

September 1997

# Specification of the Controlled-Load Network Element Service

## Status of this Memo



## Abstract

This memo specifies the network element behavior required to deliver Controlled-Load service in the Internet. Controlled-load service provides the client data flow with a quality of service closely approximating the QoS that same flow would receive from an unloaded network element, but uses capacity (admission) control to assure that this service is received even when the network element is overloaded.

## 1. Introduction

This document defines the requirements for network elements that support the Controlled-Load service. This memo is one of a series of documents that specify the network element behavior required to support various qualities of service in IP internetworks. Services described in these documents are useful both in the global Internet and private IP networks.

This document is based on the service specification template given in [1]. Please refer to that document for definitions and additional information about the specification of qualities of service within the IP protocol family.

## 2. End-to-End Behavior

The end-to-end behavior provided to an application by a series of network elements providing controlled-load service tightly approximates the behavior visible to applications receiving best-effort service *under unloaded conditions* from the same series of network elements. Assuming the network is functioning correctly, these applications may assume that:

- A very high percentage of transmitted packets will be successfully delivered by the network to the receiving end-nodes. (The percentage of packets not successfully delivered must closely approximate the basic packet error rate of the transmission medium).

- The transit delay experienced by a very high percentage of the delivered packets will not greatly exceed the minimum transmit delay experienced by any successfully delivered packet. (This minimum transit delay includes speed-of-light delay plus the fixed processing time in routers and other communications devices along the path.)

To ensure that these conditions are met, clients requesting controlled-load service provide the intermediate network elements with a estimation of the data traffic they will generate; the TSpec. In return, the service ensures that network element resources adequate to process traffic falling within this descriptive envelope will be available to the client. Should the client’s traffic generation properties fall outside of the region described by the TSpec parameters, the QoS provided to the client may exhibit characteristics indicative of overload, including large numbers of delayed or dropped packets. The service definition does not require that the precise characteristics of this overload behavior match those which would be received by a best-effort data flow traversing the same path under overloaded conditions.

NOTE: In this memo, the term "unloaded" is used in the sense of "not heavily loaded or congested" rather than in the sense of "no other network traffic whatsoever".

## 3. Motivation

The controlled load service is intended to support a broad class of applications which have been developed for use in today’s Internet, but are highly sensitive to overloaded conditions. Important members of this class are the "adaptive real-time applications" currently

offered by a number of vendors and researchers. These applications have been shown to work well on unloaded nets, but to degrade quickly under overloaded conditions. A service which mimics unloaded nets serves these applications well.

The controlled-load service is intentionally minimal, in that there are no optional functions or capabilities in the specification. The service offers only a single function, and system and application designers can assume that all implementations will be identical in this respect.

Internally, the controlled-load service is suited to a wide range of implementation techniques, including evolving scheduling and admission control algorithms that allow implementations to be highly efficient in the use of network resources. It is equally amenable to extremely simple implementation in circumstances where maximum utilization of network resources is not the only concern.

## 4. Network Element Data Handling Requirements

Each network element accepting a request for controlled-load service must ensure that adequate bandwidth and packet processing resources are available to handle the requested level of traffic, as given by the requestor's TSpec. This must be accomplished through active admission control. All resources important to the operation of the network element must be considered when admitting a request. Common examples of such resources include link bandwidth, router or switch port buffer space, and computational capacity of the packet forwarding engine.

The controlled-load service does not accept or make use of specific target values for control parameters such as delay or loss. Instead, acceptance of a request for controlled-load service is defined to imply a commitment by the network element to provide the requestor with service closely equivalent to that provided to uncontrolled (best-effort) traffic under lightly loaded conditions.

The definition of "closely equivalent to unloaded best-effort service" is necessarily imprecise. It is easiest to define this quality of service by describing the events which are expected to *not* occur with any frequency. A flow receiving controlled-load service at a network element may expect to experience:

- Little or no average packet queueing delay over all timescales significantly larger than the "burst time". The burst time is defined as the time required for the flow's maximum size data burst to be transmitted at the flow's requested transmission rate, where the burst size and rate are given by the flow's TSpec, as described below.

- Little or no congestion loss over all timescales significantly larger than the "burst time" defined above. In this context, congestion loss includes packet losses due to shortage of any required processing resource, such as buffer space or link bandwidth. Although occasional congestion losses may occur, any substantial sustained loss represents a failure of the admission control algorithm.

The basic effect of this language is to establish an expectation on the *duration* of a disruption in delivery service. Events of shorter duration are viewed as statistical effects which may occur in normal operation. Events of longer duration are indicative of failure to allocate adequate capacity to the controlled-load flow.

A network element may employ statistical approaches to decide whether adequate capacity is available to accept a service request. For example, a network element processing a number of flows with long-term characteristics predicted through measurement of past behavior may be able to overallocate its resources to some extent without reducing the level of service delivered to the flows.

A network element may employ any appropriate scheduling means to ensure that admitted flows receive appropriate service.

NOTE: The flexibility implied by the above paragraph exists within definite limits. Readers should observe that the specification's requirement that the delay and loss behavior described above imposes concrete requirements on implementations.

Perhaps the most important requirement is that the implementation has to make bandwidth greater than the Tspec token rate available to the flow in certain situations. The requirement for the availability of extra bandwidth may be derived from the fluid model of traffic scheduling (e.g. [7]). If a flow receives exactly its promised token rate at all times, queueing caused by an over-rate burst arriving at the network element may never clear, causing the traffic queueing delay to permanantly increase. This will happen if the flow continues to generate traffic at exactly the token rate after emitting the burst.

To control the long-term effects of traffic bursts, a Controlled Load implementation has several options. At minimum, a mechanism must be present to "borrow" bandwidth needed to clear bursts from the network. There are a number of ways to implement such a mechanism, ranging from explicit borrowing schemes within the traffic scheduler to implicit schemes based on statistical multiplexing and measurement-based admission control. The specification does not prefer any method over any other, but does require that some such mechanism must exist.

Similarly, the requirement for low congestion loss for in-Tspec traffic implies that buffer management must have some flexibility. Because the controlled-load service does not reshape traffic to its token-bucket parameters at every node, traffic flowing through the network will be distorted as it traverses queueing points. This distortion is particularly likely to occur during traffic bursts, precisely when buffering is most heavily used. In these circumstances, rigidly restricting the buffering capacity to a size equal to the flow's TSpec burst size may lead to congestion loss. An implementaton should be prepared to make additional buffering available to bursting flows. Again, this may be accomplished in a number of ways. One obvious choice is statistical multiplexing of a shared buffer pool.

Links are not permitted to fragment packets which receive the controlled-load service. Packets larger than the MTU of the link must be treated as nonconformant to the TSpec. This implies that they will be forwarded according to the rules described in the Policing section below.

Implementations of controlled-load service are not required to provide any control of short-term packet delay jitter beyond that described above. However, the use of packet scheduling algorithms that provide additional jitter control is not prohibited by this specification.

Packet losses due to non-congestion-related causes, such as link errors, are not bounded by this service.

## 5. Invocation Information

The controlled-load service is invoked by specifying the data flow's desired traffic parameters (TSpec) to the network element. Requests placed for a new flow will be accepted if the network element has the capacity to forward the flow's packets as described above. Requests to change the TSpec for an existing flow should be treated as a new invocation, in the sense that admission control must be reapplied to the flow. Requests that reduce the TSpec for an existing flow (in the

sense that the new TSpec is strictly smaller than the old TSpec according to the ordering rules given below) should never be denied service.

The Controlled-Load service uses the TOKEN_BUCKET_TSPEC defined in Reference [5] to describe a data flow's traffic parameters. This TSpec takes the form of a token bucket specification plus a peak rate (p), a minimum policed unit (m) and a maximum packet size (M).

The token bucket specification includes a bucket rate r and a bucket depth, b. Both r and b must be positive. The rate, r, is measured in bytes of IP datagrams per second. Values of this parameter may range from 1 byte per second to 40 terabytes per second. Network elements MUST return an error for requests containing values outside this range. Network elements MUST return an error for any request containing a value within this range which cannot be supported by the element. In practice, only the first few digits of the r parameter are significant, so the use of floating point representations, accurate to at least 0.1% is encouraged.

The bucket depth, b, is measured in bytes. Values of this parameter may range from 1 byte to 250 gigabytes. Network elements MUST return an error for requests containing values outside this range. Network elements MUST return an error for any request containing a value within this range which cannot be supported by the element. In practice, only the first few digits of the b parameter are significant, so the use of floating point representations, accurate to at least 0.1% is encouraged.

The range of values allowed for these parameters is intentionally large to allow for future network technologies. Any given network element is not expected to support the full range of values.

The peak rate, p, is measured in bytes of IP datagrams per second and has the same range and suggested representation as the bucket rate. The peak rate parameter exists in this version of the specification primarily for TSpec compatability with other QoS control services and the shared TOKEN_BUCKET_TSPEC parameter. While some admission control and buffer allocation algorithms may find the peak rate value useful, the field may always be ignored by a Controlled-Load service conforming to this version of the specification. That is, the service module at a network element may always assume that the peak data rate arriving at that element is the line rate of the incoming interface, and the service's evaluation criteria do not require a network element to consider the peak rate value. More explicit use of the peak-rate parameter by a Controlled-Load service module may be added to the specification in the future.

The minimum policed unit, m, is an integer measured in bytes. All IP datagrams less than size m will be counted against the token bucket as being of size m. The maximum packet size, M, is the biggest packet that will conform to the traffic specification; it is also measured in bytes. Network elements MUST reject a service request if the requested maximum packet size is larger than the MTU of the link. Both m and M must be positive, and m must be less then or equal to M.

The preferred concrete representation for the TSpec is three floating point numbers in single-precision IEEE floating point format followed by two 32-bit integers in network byte order. The first value is the rate (r), the second value is the bucket size (b), the third is the peak rate (p), the fourth is the minimum policed unit (m), and the fifth is the maximum packet size (M). For the parameters (r) and (b), only bit-patterns which represent valid non-negative floating point numbers are allowed. Negative numbers (including "negative zero), infinities, and NAN's are not allowed. For the parameter (p) only bit-patterns which represent valid non-negative floating point numbers or positive infinity are allowed. Positive infinity is represented with an exponent of all ones (255) and a sign bit and mantissa of all zeroes. Negative numbers (including "negative zero"), negative infinity, and NAN's are not allowed.

NOTE: An implementation which utilizes general-purpose hardware or software IEEE floating-point support may wish to verify that arriving parameters meet this requirement before using the parameters in floating-point computations, in order to avoid unexpected exceptions or traps.

The controlled-load service is assigned service_name 5.

The TOKEN_BUCKET_TSPEC parameter used by the Controlled-Load service is general parameter number 127, as indicated in [5].

## 6. Exported Information

The controlled-load service has no required characterization parameters. Individual implementations may export appropriate implementation-specific measurement and monitoring information.

## 7. Policing

The controlled-load service is provided to a flow on the basis that the flow's traffic conforms to a TSpec given at flow setup time. This section defines the meaning of conformance to the controlled-load TSpec, describes the circumstances under which a controlled-load flow's traffic might *not* conform to the TSpec, and specifies the network element's action in those circumstances.

Controlled-load service modules provide QoS control for traffic conforming to the TSpec given at setup time. The TSpec's token bucket parameters require that traffic must obey the rule that over all time periods, the amount of data sent does not exceed rT+b, where r and b are the token bucket parameters and T is the length of the time period. For the purposes of this accounting, links must count packets that are smaller than the minimal policing unit m to be of size m. Packets that arrive at an element and cause a violation of the the rT+b bound are considered nonconformant.

Additionally, packets bigger than the outgoing link MTU are considered nonconformant. It is expected that this situation will not arise with any frequency, because flow setup mechanisms are expected to notify the sending application of the appropriate path MTU.

In the presence of nonconformant packets arriving for one or more controlled-load flows, each network element must ensure locally that the following requirements are met:

1) The network element MUST continue to provide the contracted quality of service to those controlled-load flows not experiencing excess traffic.

2) The network element SHOULD prevent excess controlled-load traffic from unfairly impacting the handling of arriving best-effort traffic. This requirement is discussed further in Section 9 of this document (Guidelines for Implementors).

3) Consistent with points 1 and 2, the network element MUST attempt to forward the excess traffic on a best-effort basis if sufficient resources are available.

Network elements must not assume that that arrival of nonconformant traffic for a specific controlled-load flow will be unusual, or indicative of error. In certain circumstances (particularly, routers acting as the "split points" of a multicast distribution tree supporting a shared reservation) large numbers of packets will fail the conformance test *as a matter of normal operation*.

Network elements must not assume that data sources or upstream elements have taken action to "police" controlled-load flows by limiting their traffic to conform to the flow's TSpec. Each network element providing controlled-load service MUST independently ensure that the requirements given above are met in the presence of nonconformant arriving traffic for one or more controlled-load flows.

Network elements may use any appropriate implementation mechanism to meet the requirements given above. Examples of such mechanisms include token-bucket policing filters and per-flow scheduling algorithms. However, it is insufficient to simply place all controlled-load flows into the same shared resource pool, without first ensuring that non-conformant flows are prevented from starving conformant flows of the necessary processing resources.

Further discussion of this issue may be found in Section 11 of this note.

Beyond requirements 2 and 3 above, the controlled-load service does not define the QoS behavior delivered to flows with non-conformant arriving traffic. Specifically, it is permissible either to degrade the service delivered to all of the flow's packets equally, or to sort the flow's packets into a conformant set and a nonconformant set and deliver different levels of service to the two sets. This point is discussed further in Section 9 of this note.

When resources are available, network elements at points within the interior of the network SHOULD be prepared to accommodate packet bursts somewhat larger than the actual TSpec. This requirement derives from the traffic distortion effect described in Section 4. As described there, it may be met either through explicit means or statistical multiplexing of shared buffering resources.

When handling such traffic, it is permissible to allow some delaying of a packet if that delay would allow it to pass the policing function. (In other words, to reshape the traffic). However, the overall requirement for limiting the duration of any such traffic distortion must be considered. The challenge is to define a viable reshaping function.

Intuitively, a plausible approach is to allow a delay of (roughly) up to the maximum queueing delay experienced by completely conforming packets before declaring that a packet has failed to pass the policing function. The merit of this approach, and the precise wording of the specification that describes it, require further study.

## 8. Ordering and Merging

The controlled-load service TSpec is ordered according to the following rule: TSpec A is a substitute for ("as good or better than" or "greater than or equal to") TSpec B if and only if:

(1) the token bucket rate r for TSpec A is greater than or equal to that of TSpec B,

(2) the token bucket depth b for TSpec A is greater than or equal to that of TSpec B,

(3) the peak rate p for TSpec A is greater than or equal to that of TSpec B,

(4) the minimum policed unit m for TSpec A is less than or equal to that of TSpec B,

(5) the maximum packet size M of TSpec A is greater than or equal to that of TSpec B.

Note that not all TSpecs can be ordered with respect to each other. If two TSpecs differ but not all five of the points above are true, then the TSpecs are unordered.

A merged TSpec is the TSpec used by the RSVP protocol when merging a set of TSpecs to create a "merged" reservation. TSpec merging is described further in [4] and [3]. The TSpec merge operation addresses two requirements:

- The "merged" TSpec parameters are used as the traffic flow's TSpec at the local node.

- The merged parameters are passed upstream to traffic source(s) to describe characteristics of the actually installed reservation along the data path.

For the controlled-load service, a merged TSpec may be calculated over a set of TSpecs by taking:

(1) the largest token bucket rate r;

(2) the largest token bucket size b;

(3) the largest peak rate p;

(4) the smallest minimal policed unit m;

(5) the *smallest* maximum packet size M;

across all members of the set.

A Least Common TSpec is a TSpec adequate to describe the traffic from any one of a number of traffic flows. The least common TSpec may be useful when creating a shared reservation for a number of flows using SNMP or another management protocol. This differs from the merged TSpec described above in that the computed parameters are not passed upstream to the sources of traffic.

For the controlled-load service, the Least Common TSpec may be calculated over a set of TSpecs by taking:

(1) the largest token bucket rate r;

(2) the largest token bucket size b;

(3) the largest peak rate p;

(4) the smallest minimal policed unit m;

(5) the largest maximum packet size M;

across all members of the set.

The sum of n controlled-load service TSpecs is used when computing the TSpec for a shared reservation of n flows. It is computed by taking:

- The sum across all TSpecs of the token bucket rate parameter r.

- The sum across all TSpecs of the token bucket size parameter b.

- The sum across all TSpecs of the peak rate parameter p.

- The minimum across all TSpecs of the minimum policed unit parameter m.

- The maximum across all TSpecs of the maximum packet size parameter M.

The minimum of two TSpecs differs according to whether the TSpecs can be ordered according to the "greater than or equal to" rule above. If one TSpec is less than the other TSpec, the smaller TSpec is the minimum. For unordered TSpecs, a different rule is used. The minimum of two unordered TSpecs is determined by comparing the respective values in the two TSpecs and choosing:

(1) the smaller token bucket rate r;

(2) the *larger* token bucket size b;

(3) the smaller peak rate p;

(4) the *smaller* minimum policed unit m;

(5) the smaller maximum packet size M;

## 9. Guidelines for Implementors

REQUIREMENTS PLACED ON ADMISSION CONTROL ALGORITHM: The intention of this service specification is that network elements deliver a level of service closely approximating best-effort service under unloaded conditions. As with best-effort service under these conditions, it is not required that every single packet must be successfully delivered with zero queueing delay. Network elements providing controlled-load service are permitted to oversubscribe the available resources to some extent, in the sense that the bandwidth and buffer requirements indicated by summing the TSpec token buckets of all controlled-load flows may exceed the maximum capabilities of the network element. However, this oversubscription may only be done in cases where the element is quite sure that actual utilization is less than the sum of the token buckets would suggest, so that the implementor's performance goals will be met. This information may come from measurement of the aggregate traffic flow, specific knowledge of application traffic statistics, or other means. The most conservative approach, rejection of new flows whenever the addition of their traffic would cause the strict sum of the token buckets to exceed the capacity of the network element (including consideration of resources needed to maintain the delay and loss characteristics specified by the service) may be appropriate in other circumstances.

Specific issues related to this subject are discussed in the "Evaluation Criteria" and "Examples of Implementation" sections below.

INTERACTION WITH BEST-EFFORT TRAFFIC: Implementors of this service should clearly understand that in certain circumstances (routers acting as the "split points" of a multicast distribution tree supporting a shared reservation) large numbers of a flow's packets may fail the TSpec conformance test *as a matter of normal operation*. According to the requirements of Section 7, these packets should be forwarded on a best-effort basis if resources permit.

If the network element's best-effort queueing algorithm does not distinguish between these packets and elastic best-effort traffic such as TCP flows, THE EXCESS CONTROLLED-LOAD PACKETS WILL "BACK OFF" THE ELASTIC TRAFFIC AND DOMINATE THE BEST-EFFORT BANDWIDTH USAGE. The integrated services framework does not currently address this issue. However, several possible solutions to the problem are known [RED, xFQ]. Network elements supporting the controlled load service should implement some mechanism in their best-effort queueing path to discriminate between classes of best-effort traffic and provide elastic traffic with protection from inelastic best-effort flows.

Two basic approaches are available to meet this requirement. The network element can maintain separate resource allocations for different classes of best-effort traffic, so that no one class will excessively dominate the loaded best-effort mix. Alternatively, an element can process excess controlled-load traffic at somewhat lower priority than elastic best-effort traffic, so as to completely avoid the back-off effect discussed above.

If most or all controlled-load traffic arises from non-rate-adaptive real-time applications, the use of priority mechanisms might be desirable. If most controlled-load traffic arises from rate-adaptive realtime or elastic applications attempting to establish a bounded minimum level of service, the use of separate resource classes might be preferable. However, this is not a firm guideline. In practice, the network element designer's choice of mechanism will depend heavily on both the goals of the design and the implementation techniques appropriate for the designer's platform. This version of the service specification does not specify one or the other behavior, but leaves the choice to the implementor.

FORWARDING BEHAVIOR IN PRESENCE OF NONCONFORMANT TRAFFIC: As indicated in Section 7, the controlled-load service does not define the QoS behavior delivered to flows with non-conformant arriving traffic. It is permissible either to degrade the service delivered to all of the flow's packets equally, or to sort the flow's packets into a conformant set and a nonconformant set and deliver different levels of service to the two sets.

In the first case, expected queueing delay and packet loss probability will rise for all packets in the flow, but packet delivery reordering will, in general, remain at low levels. This behavior is preferable for those applications or transport protocols which are sensitive to excessive packet reordering. A possible example is an unmodified TCP connection, which would see reordering as lost packets, triggering duplicate acks and hence excessive retransmissions.

In the second case, some subset of the flow's packets will be delivered with low loss and delay, while some other subset will be delivered with higher loss and potentially higher delay. The delayed packets will appear to the receiver to have been reordered in the network, while the non-delayed packets will, on average, arrive in a more timely fashion than if all packets were treated equally. This might be preferable for applications which are highly time-sensitive, such as interactive conferencing tools.

## 10. Evaluation Criteria

The basic requirement placed on an implementation of controlled-load service is that, under all conditions, it provide accepted data flows with service closely similar to the service that same flow would receive using best-effort service under unloaded conditions.

This suggests a simple two-step evaluation strategy. Step one is to compare the service given best-effort traffic and controlled-load traffic under underloaded conditions.

- Measure the packet loss rate and delay characteristics of a test flow using best-effort service and with no load on the network element.

- Compare those measurements with measurements of the same flow receiving controlled-load service with no load on the network element.

  Closer measurements indicate higher evaluation ratings. A substantial difference in the delay characteristics, such as the smoothing which would be seen in an implementation which scheduled the controlled-load flow using a fixed, constant-bitrate algorithm, should result in a somewhat lower rating.

Step two is to observe the change in service received by a controlled-load flow as the load increases.

- Increase the background traffic load on the network element, while continuing to measuring the loss and delay characteristics of the controlled-load flow. Characteristics which remain essentially constant as the element is driven into overload indicate a high evaluation rating. Minor changes in the delay distribution indicate a somewhat lower rating. Significant increases in delay or loss indicate a poor evaluation rating.

This simple model is not adequate to fully evaluate the performance of controlled-load service. Three additional variables affect the evaluation. The first is the short-term burstiness of the traffic stream used to perform the tests outlined above. The second is the degree of long-term change in the controlled-load traffic within the bounds of its TSpec. (Changes in this characteristic will have great effect on the effectiveness of certain admission control algorithms.) The third is the ratio of controlled-load traffic to other traffic at the network element (either best effort or other controlled services).

The third variable should be specifically evaluated using the following procedure.

With no controlled-load flows in place, overload the network element with best-effort traffic (as indicated by substantial packet loss and queueing delay).

Execute requests for controlled-load service giving TSpecs with increasingly large rate and burst parameters. If the request is accepted, verify that traffic matching the TSpec is in fact handled with characteristics closely approximating the unloaded measurements taken above.

Repeat these experiments to determine the range of traffic parameter (rate, burst size) values successfully handled by the network element. The useful range of each parameter must be determined for several settings of the other parameter, to map out a two-dimensional "region" of successfully handled TSpecs. When compared with network elements providing similar capabilities, this region indicates the relative ability of the elements to provide controlled-load service under high load. A larger region indicates a higher evaluation rating.

## 11. Examples of Implementation

One possible implementation of controlled-load service is to provide a queueing mechanism with two priority levels; a high priority one for controlled-load and a lower priority one for best effort service. An admission control algorithm is used to limit the amount of traffic placed into the high-priority queue. This algorithm may be based either on the specified characteristics of the high-priority flows (using information provided by the TSpecs), or on the measured characteristics of the existing high-priority flows and the TSpec of the new request.

Another possible implementation of controlled-load service is based on the existing capabilities of network elements which support

"traffic classes" based on mechanisms such as weighted fair queueing or class-based queueing [6]. In this case, it is sufficient to map data flows accepted for controlled-load service into an existing traffic class with adequate capacity to avoid overload. This

requirement is enforced by an admission control algorithm which considers the characteristics of the traffic class, the characteristics of the traffic already admitted to the class, and the TSpec of the new flow requesting service. Again, the admission control algorithm may be based either on the TSpec-specified or the measured characteristics of the existing traffic.

A specific case of the above approach is to employ a scheduler which implements weighted fair queueing or similar load-management scheme, allocating a separate scheduling queue with correctly chosen weight to each individual controlled-load flow. In this circumstance, the traffic scheduler also plays the role of the policing function, by ensuring that nonconformant traffic arriving for one controlled-load flow does not affect either other controlled-load flows or the best-effort traffic. This elimination of mechanism is balanced by the drawback that the approach does not benefit from any performance or resource usage gain arising from statistical aggregation of several flows into a single queueing class.

Admission control algorithms based on specified characteristics are likely be appropriate when the number of flows in the high-priority class is small, or the traffic characteristics of the flows appear highly variable. In these situations the measured behavior of the aggregate controlled-load traffic stream may not serve as an effective predictor of future traffic, leading a measurement-based admission control algorithm to produce incorrect results. Conversely, in situations where the past behavior of the aggregate controlled-load traffic *is* a good predictor of future behavior, a measurement-based admission control algorithm may allow more traffic to be admitted to the controlled-load service class with no degradation in performance. An implementation may choose to switch between these two approaches depending on the nature of the traffic stream at a given time.

A variety of techniques may be used to provide the desired isolation between excess (nonconformant) controlled-load traffic and other best-effort traffic. Use of a low priority queue for nonconformant controlled-load traffic is simple, but other approaches may provide superior service or fit better into existing architectures. Variants of fair queueing or weighted fair queueing may be used to allocate a percentage of the available resources to different best-effort traffic classes. One approach would be to allocate each controlled-load flow a a 1/N "fair share" percentage of the available best-

effort bandwidth for its excess traffic. An alternate approach would be to provide a single WFQ resource class for all excess controlled-load traffic. Finally, alternate mechanisms such as RED [xxx] may be used to provide the same overall function.

## 12. Examples of Use

The controlled-load service may be used by any application which can make use of best-effort service, but is best suited to those applications which can usefully characterize their traffic requirements. Applications based on the transport of "continuous media" data, such as digitized audio or video, are an important example of this class.

The controlled-load service is not isochronous and does not provide any explicit information about transmission delay. For this reason, applications with end-to-end timing requirements, including the continuous-media class mentioned above, provide an application-specific timing recovery mechanism, similar or identical to the mechanisms required when these applications use best-effort service. A protocol useful to applications requiring this capability is the IETF Real-Time Transport Protocol [2].

Load-sensitive applications may choose to request controlled-load service whenever they are run. Alternatively, these applications may monitor their own performance and request controlled-load service from the network only when best-effort service is not providing acceptable performance. The first strategy provides higher assurance that the level of quality delivered to the user will not change over the lifetime of an application session. The second strategy provides greated flexibility and offers cost savings in environments where levels of service above best-effort incur a charge.

## 13. Security Considerations

A network element implementing the service described here is intentionally and explicitly expected to give preferential treatment to selected packet traffic. This memo does not describe the mechanism used to indicate which traffic is to receive the preferential treatment - rather, the controlled-load service described here may be invoked by a number of mechanisms, including RSVP, SNMP network management software, or proprietary control software. However, any mechanism used to invoke the controlled load service must provide security sufficient to guard against use of this preferential treatment capability by undesired or unauthorized traffic. A correct implementation of the controlled-load service is *not* susceptable to a denial-of-service attack based on maliciously requesting a very small resource allocation for the attacked traffic flow. This is

because the service specification requires that traffic in excess of the requested level be carried on a best-effort basis, rather than being dropped. This requirement is discussed further in Section 7 of this memo.

Of necessity, giving preferential service to certain traffic flows implies giving less service to other traffic flows. Thus, it is possible to conduct a denial of service attack by maliciously reconfiguring the controlled-load "admission control algorithm" to allow overallocation of available bandwidth or other forwarding resources, starving non-controlled-load flows. In general, this is unlikely to increase the network's vulnerability to attack, because many other reconfigurations of a router or host can cause denial of service. It is reasonable to assume that whatever means is used to protect against other reconfiguration attacks will be adequate to protect against this one as well.

## Appendix 1: Use of the Controlled-Load service with RSVP

The use of Controlled-Load service in conjunction with the RSVP resource reservation setup protocol is specified in reference [4]. This document gives the format of RSVP FLOWSPEC, SENDER_TSPEC, and ADSPEC objects needed to support applications desiring Controlled-Load service and gives information about how RSVP processes those objects. The RSVP protocol itself is specified in Reference [3].

## References

[1] Shenker, S., and J. Wroclawski. "Network Element Service Specification Template", RFC 2216, September 1997.

[2] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson. "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[3] Braden, R., Ed., et. al., "Resource Reservation Protocol (RSVP) - Version 1 Functional Specification", RFC 2205, September 1997.

[4] Wroclawski, J., "The use of RSVP with IETF Integrated Services", RFC 2210, September 1997.

[5] Shenker, S., and J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", RFC 2215, September 1997.

[6] S. Floyd, and V. Jacobson. "Link-sharing and Resource Management Models for Packet Networks," IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, August 1995.

[7] A. K. J. Parekh. "A Generalized Processor Sharing Approach to Flow Control in Integrated Service Networks". MIT Laboratory for Information and Decision Systems, Report LIDS-TH-2089, February 1992

Author's Address

John Wroclawski
MIT Laboratory for Computer Science
545 Technology Sq.
Cambridge, MA 02139

Phone: 617-253-7885
Fax: 617-253-2673 (FAX)
EMail: jtw@lcs.mit.edu