Network Working Group
Request for Comments: 2215
Category: Standards Track

S. Shenker
J. Wroclawski
Xerox PARC/MIT LCS

September 1997

General Characterization Parameters for Integrated Service Network Elements

Status of this Memo

Abstract

This memo defines a set of general control and characterization parameters for network elements supporting the IETF integrated services QoS control framework. General parameters are those with common, shared definitions across all QoS control services.

1. Introduction

This memo defines the set of general control and characterization parameters used by network elements supporting the integrated services framework. "General" means that the parameter has a common definition and shared meaning across all QoS control services.

Control parameters are used by applications to provide information to the network related to QoS control requests. An example is the traffic specification (TSpec) generated by application senders and receivers.

Characterization parameters are used to discover or characterize the QoS management environment along the path of a packet flow requesting active end-to-end QoS control. These characterizations may eventually be used by the application requesting QoS control, or by other network elements along the path. Examples include information about which QoS control services are available along a network path and estimates of the available path bandwidth.

Individual QoS control service specifications may refer to these parameter definitions as well as defining additional parameters specific to the needs of that service.

Parameters are assigned machine-oriented ID's using a method described in [RFC 2216] and summarized here. These ID's may be used within protocol messages (e.g., as described in [RFC 2210]) or management interfaces to describe the parameter values present. Each parameter ID is composed from two numerical fields, one identifying the service associated with the parameter (the <service_number>), and the other (the <parameter_number>) identifying the parameter itself. Because the definitions of the parameters defined in this note are common to all QoS control services, the <parameter_number> values for the parameters defined here are assigned from the "general parameters" range (1 - 127).

NOTE: <parameter_numbers> in the range 128 - 254 name parameters with definitions specific to a particular QoS control service. In contrast to the general parameters described here, it is necessary to consider both the <service_number> and <parameter_number> to determine the meaning of the parameter.

Service number 1 is reserved for use as described in Section 2 of this note. Service numbers 2 through 254 will be allocated to individual QoS control services. Currently, Guaranteed service [RFC 2212] is allocated number 2, and Controlled-load service [RFC 2211] is allocated number 5.

In this note, the textual form

<service_number, parameter_number>

is used to write a service_number, parameter_number pair. The range of possible of service_number and parameter_number values specified in [RFC 2216] allow the parameter ID to directly form the tail portion of a MIB object ID representing the parameter. This simplifies the task of making parameter values available to network management applications.

The definition of each parameter used to characterize a path through the network describes two types of values; local and composed. A Local value gives information about a single network element. Composed values reflect the running composition of local values along a path, specified by some composition rule. Each parameter definition specifies the composition rule for that parameter. The composition rule tells how to combine an incoming composed value (from the already-traversed portion of the path) and the local value, to give a new composed value which is passed to the next network element in the path. Note that the composition may proceed either

downstream, toward the receiver(s), or upstream, toward the sender. Each parameter may give only one definition for the local value, but may potentially give more than one definition for composition rules and composed values. This is because it may be useful to compose the same local value several times following different composition rules.

Because characterization parameters are used to compute the properties of a specific path through the internetwork, all characterization parameter definitions are conceptually "per-next-hop", as opposed to "per interface" or "per network element". In cases where the network element is (or is controlling) a shared media or large-cloud subnet, the element may need to provide different values for different next-hops within the cloud. In practice, it may be appropriate for vendors to choose and document a tolerance range, such that if all next-hop values are within the tolerance range only a single value need be stored and provided.

Local and composed characterization parameter values have distinct ID's so that a network management entity can examine the value of either a local or path-composed parameter at any point within the network.

Each parameter definition includes a description of the minimal properties, such as range and precision, required of any wire representation of that parameter's values. Each definition also includes an XDR [RFC 1832] description of the parameter, describing an appropriate external (wire) data representation for the parameter's values. This dual definition is intended to encourage a common wire representation format whenever possible, while still allowing other representations when required by the specific circumstances (e.g., ASN.1 within SNMP).

The message formats specified in [RFC 2210] for use with the RSVP setup protocol use the XDR data representation parameters.

All of the parameters described in this note are mandatory, in the sense that a network element claiming to support integrated service must recognize arriving values in setup and management protocol messages, process them correctly, and export a reasonable value in response. For some parameters, the specification requires that the network element compute and export an *accurate* local value. For other parameters, it is acceptable for the network element to indicate that it cannot compute and export an accurate local value. The definition of these parameters provides a reserved value which indicates "indeterminate" or "invalid". This value signals that an element cannot process the parameter accurately, and consequently that the result of the end-to-end composition is also questionable.

NOTE (temporary): Previous versions of this and the RSVP use document used both the reserved-value approach and a separate INVALID flag to record this fact. Now, the reserved-value approach is used exclusively. This is so that any protocol which retrieves a parameter value, including SNMP, can carry the invalid indication without needing a separate flag. The INVALID flag remains in the RSVP message format but is reserved for use only with a possible future service-composition scheme.

2. Default and Service-Specific Values for General Parameters

General parameters have a common *definition* across all QoS control services. Frequently, the same *value* of a general parameter will be correct for all QoS control services offered by a network element. In this circumstance, there is no need to export a separate copy of the value for each QoS control service; instead the node can export one number which applies to all supported services.

A general parameter value which applies to all services supported at a network node is called a default or global value. For example, if all of the QoS control services provided at a node support the same maximum packet size, the node may export a single default value for the PATH_MTU parameter described in Section 3, rather than providing a separate copy of the value for each QoS control service. In the common case, this reduces both message size and processing overhead for the setup protocol.

Occasionally an individual service needs to report a value differing from the default value for a particular general parameter. For example, if the implementation of Guaranteed Service [RFC 2212] at a router is restricted by scheduler or hardware considerations to a maximum packet size smaller than supported by the router's best-effort forwarding path, the implementation may wish to export a "service-specific" value of the PATH_MTU parameter so that applications using the Guaranteed service will function correctly.

In the example above, the router might supply a value of 1500 for the default PATH_MTU parameter, and a value of 250 for the PATH_MTU parameter applying to guaranteed service. In this case, the setup protocol providing path characterization carries (and delivers to the application) both a value for Guaranteed service and a value for other services.

The distinction between default and service-specific parameter values makes no sense for non-general parameters (those defined by a specific QoS control service, rather than this note), because both the definition and value of the parameter are always specific to the particular service.

The distinction between default and service-specific values for general parameters is reflected in the parameter ID name space. This allows network nodes, setup protocols, and network management tools to distinguish default from service-specific values, and to determine which service a service-specific parameter value is associated with.

Service number 1 is used to indicate the default value. A parameter value identified by the ID:

<1, parameter_number>

is a default value, which applies to all services unless it is overridden by a service-specific value for the same parameter.

A parameter value identified by the ID:

<service_number, parameter_number>

where service_number is not equal to 1, is a service-specific value. It applies only to the service identified by service_number.

These service-specific values are also called override values. This is because when both service-specific and default values are present for a parameter, the service-specific value overrides the default value (for the service to which it applies). The rules for composing service-specific and global general parameters support this override capability. The basic rule is to use the service-specific value if it exists, and otherwise the global value.

A complete summary of the characterization parameter composition process is given below. In this summary, the "arriving value" is the incompletely composed parameter value arriving from a neighbor node. The "local value" is the (global or service-specific) value made available by the local node. The "result" is the newly composed value to be sent to the next node on the data path.

1. Examine the <service_number, parameter_number> pair associated with the arriving value. This information is conveyed by the setup protocol together with the arriving value.

2. If the arriving value is for a parameter specific to a single service (this is true when the parameter_number is larger than 128), compose the arriving value with the local value exported by the specified service, and pass the result to the next hop. In this case there is no need to consider global values, because the parameter itself is specific to just one service.

3. If the arriving value is a service-specific value for a generally defined parameter (the parameter_number is 127 or less, and the service_number is other than 1), and the local implementation of that service also exports a service-specific value for the parameter, compose the service-specific arriving value and the service-specific local value of the parameter, and pass the result as a service-specific value to the next-hop node.

4. If the arriving value is a service-specific value for a general parameter (the parameter_number is 127 or less, and the service_number is other than 1), and the local implementation of that service does *not* export a service-specific value, compose the service-specific arriving value with the global value for that parameter exported by the local node, and pass the result as a service-specific value to the next-hop node.

5. If the arriving value is a global value for a general parameter (parameter_number is 127 or less, and the service_number is 1), and the local implementation of *any* service exports a service-specific value for that general parameter, compose the arriving (global) value with the service-specific value for that parameter exported by the local service, and pass the result as a service-specific value to the next-hop node. This will require adding a new data field to the message passed to the next hop, to hold the newly generated service-specific value. Repeat this process for each service that exports a service-specific value for the parameter.

6. If the arriving value is a global value for a general parameter (the service_number is 1, and the parameter_number is 127 or less), compose the arriving (global) value with the global parameter value exported by the local node, and pass the result as a global (service 1) value to the next-hop node. This step is performed whether or not any service-specific values were generated and exported in step 5.

3. General Parameter Definitions

3.1 NON-IS_HOP flag parameter

This parameter provides information about the presence of network elements which do not implement QoS control services along the data path.

The local value of the parameter is 1 if the network element does not implement the relevant QoS control service, or knows that there is a break in the chain of elements which implement the service. The local parameter is 0 otherwise. The local parameter is assigned parameter_number 1.

The composition rule for this parameter is the OR function. A composed parameter value of 1 arriving at the endpoint of a path indicates that at least one point along the path does not offer the indicated QoS control service. The parameter_number for the composed quantity is 2.

The global NON_IS_HOP flag parameter thus has the ID <1,2>. If this flag is set, it indicates that one or more network elements along the application's data path does not support the integrated services framework at all. An example of such an element would be an IP router offering only best-effort packet delivery and not supporting any resource reservation requests.

Obviously, a network element which does not support this specification will not know to set this flag. The actual responsibility for determining that a network node does not support integrated services may fall to the network element, the setup protocol, or a manual configuration operation and is dependent on implementation and usage. This calculation must be conservative. For example, a router sending packets into an IP tunnel must assume that the tunneled packets will not receive QoS control services unless it or the setup protocol can prove otherwise.

Service-specific versions of the NON_IS_HOP flag indicate that one or more network elements along a path don't support the particular service. For example, the flag parameter identified by ID <2,2> being set indicates that some network element along the path does not support the Guaranteed service, though it might support another service such as Controlled-Load.

If the global NON_IS_HOP flag <1,2> is set for a path, the receiver (network element or application) should consider the values of all other parameters defined in this specification, including service-specific NON_IS_HOP flags, as possibly inaccurate. If a service specific NON_IS_HOP flag is set for a path, the receiver should consider the values of all other parameters associated with that service as possibly inaccurate.

The NON_IS_HOP parameter may be represented in any form which can express boolean true and false. However, note that a network element must set this flag precisely when it does *not* fully understand the format or data representation of an arriving protocol message (because it does not support the specified service). Therefore, the data representation used for this parameter by setup and management protocols must allow the parameter value to be read and set even if the network element cannot otherwise parse the protocol message.

An appropriate XDR description of this parameter is:

```
bool NON_IS_HOP;
```

However, the standard XDR data encoding for this description will not meet the requirement described above unless other restrictions are placed on message formats. An alternative data representation may be more appropriate.

NOTE: The message format described for RSVP in [RFC 2210] carries this parameter as a single-bit flag, referred to as the "break bit".

3.2 NUMBER_OF_IS_HOPS

IS stands for "integrated services aware". An integrated services aware network element is one that conforms to the various requirements described in this and other referenced documents. The network element need not offer a specific service, but if it does it must support and characterize the service in conformance with the relevant specification, and if it does not it must correctly set the NON_IS_HOP flag parameter for the service. For completeness, the local parameter is assigned the parameter_number 3.

The composition rule for this parameter is to increment the counter by one at each IS-aware hop. This quantity, when composed end-to-end, informs the endpoint of the number of integrated-services aware network elements traversed along the path. The parameter_number for this composed parameter is 4.

Values of the composed parameter will range from 1 to 255, limited by the bound on IP hop count.

The XDR representation of this parameter is:

```
unsigned int NUMBER_OF_IS_HOPS;
```

3.3. AVAILABLE_PATH_BANDWIDTH

This parameter provides information about the bandwidth available along the path followed by a data flow. The local parameter is an estimate of the bandwidth the network element has available for packets following the path. Computation of the value of this parameter should take into account all information available to the network element about the path, taking into consideration administrative and policy controls on bandwidth, as well as physical resources.

NOTE: This parameter should reflect, as closely as possible, the actual bandwidth available to packets following a path. However, the bandwidth available may depend on a number of factors not known to the network element until a specific QoS request is in place, such as the destination(s) of the packet flow, the service to be requested by the flow, or external policy information associated with a reservation request. Because the parameter must in fact be provided before any specific QoS request is made, it is frequently difficult to provide the parameter accurately. In circumstances where the parameter cannot be provided accurately, the network element should make the best attempt possible, but it is acceptable to overestimate the available bandwidth by a significant amount.

The parameter_number for AVAILABLE_PATH_BANDWIDTH is 5. The global parameter <1, 5> is an estimate of the bandwidth available to any packet following the path, without consideration of which (if any) QoS control service the packets may be subject to.

In cases where a particular service is administratively or technically restricted to a limited portion of the overall available bandwidth, the service module may wish to export an override parameter which specifies this smaller bandwidth value.

The composition rule for this parameter is the MIN function. The composed value is the minimum of the network element's value and the previously composed value. This quantity, when composed end-to-end, informs the endpoint of the minimal bandwidth link along the path from sender to receiver. The parameter_number for the composed minimal bandwidth along the path is 6.

Values of this parameter are measured in bytes per second. The representation must be able to express values ranging from 1 byte per second to 40 terabytes per second, about what is believed to be the maximum theoretical bandwidth of a single strand of fiber.

Particularly for large bandwidths, only the first few digits are significant, so the use of a floating point representation, accurate to at least 0.1%, is encouraged.

The XDR representation for this parameter is:

```
float AVAILABLE_PATH_BANDWIDTH;
```

For values of this parameter only valid non-negative floating point numbers are allowed. Negative numbers (including "negative zero"), infinities, 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 parameter values meet these requirements before using the values in floating-point computations, in order to avoid unexpected exceptions or traps.

If the network element cannot or chooses not to provide an estimate of path bandwidth, it may export a local value of zero for this parameter. A network element or application receiving a composed value of zero for this parameter must assume that the actual bandwidth available is unknown.

3.4 MINIMUM_PATH_LATENCY

The local parameter is the latency of the packet forwarding process associated with the network element, where the latency is defined to be the *smallest* possible packet delay added by the network element. This delay results from speed-of-light propagation delay, from packet processing limitations, or both. It does not include any variable queuing delay which may be present.

The purpose of this parameter is to provide a baseline minimum path latency for use with services which provide estimates or bounds on additional path delay, such as Guaranteed [RFC 2212]. Together with the queuing delay bound offered by Guaranteed and similar services, this parameter gives the application knowledge of both the minimum and maximum packet delivery delay. Knowing both the minimum and maximum latency experienced by data packets allows the receiving application to accurately compute its de-jitter buffer requirements.

Note that the quantity characterized by this parameter is the absolute smallest possible value for the packet processing and transmission latency of the network element. This value is the quantity required to provide the end hosts with jitter bounds. The parameter does *not* provide an upper-bound estimate of minimum latency, which might be of interest for best-effort traffic and QoS control services which do not explicitly offer delay bounds. In other words, the parameter will always underestimate, rather than overestimate, latency, particularly in multicast and large cloud situations.

When packets traversing a network element may experience different minimal latencies over different paths, this parameter should, if possible, report an accurate latency value for each path. For example, when an ATM point-multipoint virtual circuit is used to implement IP multicast, the mechanism that implements this parameter for the ATM cloud should ideally compute a separate value for each destination. Doing this may require cooperation between the ingress

and egress elements bounding the multi-access communication cloud. The method by which this cooperation is achieved, and the choice of which IP-level network element actually provides and composes the value, is technology-dependent.

An alternative choice is to provide the same value of this parameter for all paths through the cloud. The value reported must be the smallest latency for any possible path. Note that in this situation, QoS control services (e.g., Guaranteed) which provide an upper bound on latency cannot simply add their queuing delay to the value computed by this parameter; they must also compensate for path delays above the minimum. In this case the range between the minimum and maximum packet delays reported to the application may be larger than actually occurs, because the application will be told about the minimum delay along the shortest path and the maximum delay along the actual path. This is acceptable in most situations.

A third alternative is to report the "indeterminate" value, as specified below. In this circumstance the client application may either deduce a minimum path latency through measurement, or assume a value of zero.

The composition rule for this parameter is summation with a clamp of (2**32 - 1) on the maximum value. This quantity, when composed end-to-end, informs the endpoint of the minimal packet delay along the path from sender to receiver. The parameter_number for the latency of the network element's link is 7. The parameter_number for the cumulative latency along the path is 8.

The latencies are reported in units of one microsecond. An individual element can advertise a latency value between 1 and 2**28 (somewhat over two minutes) and the total latency added across all elements can range as high as (2**32)-2. If the sum of the different elements delays exceeds (2**32)-2, the end-to-end advertised delay should be reported as indeterminate. This is described below.

Note that while the granularity of measurement is microseconds, a conforming element is free to actually measure delays more loosely. The minimum requirement is that the element estimate its delay accurately to the nearest 100 microsecond granularity. Elements that can measure more accurately are, of course, encouraged to do so.

> NOTE: Measuring in milliseconds is not acceptable, because if the minimum delay value is a millisecond, a path with several hops will lead to a composed delay of at least several milliseconds, which is likely to be misleading.

The XDR description of this parameter is:

```
unsigned int MINIMUM_PATH_LATENCY;
```

The distinguished value (2**32)-1 is taken to mean "indeterminate latency". A network element which cannot accurately predict the latency of packets it is processing should set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path latency is not known. This may happen because one or more network elements could not supply a value, or because the range of the composition calculation was exceeded.

3.5. PATH_MTU

This parameter computes the maximum transmission unit (MTU) for packets following a data path. This value is required to invoke QoS control services which require that IP packet size be strictly limited to a specific MTU. Existing MTU discovery mechanisms cannot be used because they provide information only to the sender and they do not directly allow for QoS control services to specify MTU's smaller than the physical MTU.

The local characterization parameter is the IP MTU, where the MTU of a network element is defined to be the maximum transmission unit the network element can accommodate without fragmentation, including IP and upper-layer protocol headers but not including link level headers. The composition rule is to take the minimum of the network element's MTU and the previously composed value. This quantity, when composed end-to-end, informs the endpoint of the maximum transmission unit that can traverse the path from sender to receiver without fragmentation. The parameter_number for the MTU of the network element's link is 9. The parameter_number for the composed MTU along the path is 10.

A correct and valid value of this parameter must be provided by all IS-aware network elements.

A specific service module may specify an MTU smaller than that of the overall network element by overriding this parameter with one giving the service's MTU value. A service module may not specify an MTU value larger than that given by the global parameter.

Values of this parameter are measured in bytes. The representation must be able to express values ranging from 1 byte to 2**32-1 bytes.

The XDR description of this parameter is:

```
unsigned int PATH_MTU;
```

3.6. TOKEN_BUCKET_TSPEC

This parameter is used to describe data traffic parameters using a simple token bucket filter. This parameter is used by data senders to describe the traffic parameters of traffic it expects to generate, and by QoS control services to describe the parameters of traffic for which the reservation should apply. It is defined as a general rather than service-specific parameter because the same traffic description may be used by several QoS control services in some situations.

> NOTE: All previous definitions in this note have described "characterization parameters", with local values set by network elements to characterize their behavior and composition rules to give the resulting end-to-end behavior. The TOKEN_BUCKET_TSPEC is not a characterization parameter, because intermediate nodes within the network do not export local values for TOKEN_BUCKET_TSPECs. The TOKEN_BUCKET_TSPEC is simply a data structure definition given here because it is common to more than one QoS control service.

The TOKEN_BUCKET_TSPEC parameter is assigned parameter_number 127.

The TOKEN_BUCKET_TSPEC takes the form of a token bucket specification plus a peak rate [p], minimum policed unit [m], and a maximum packet size [M].

The token bucket specification includes an average or token rate [r] and a bucket depth [b]. Both [r] and [b] must be positive.

The token 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. In practice, only the first few digits of the [r] and [p] parameters 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. 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 peak traffic rate [p] 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. In practice, only the first few digits of

the [r] and [p] parameters are significant, so the use of floating point representations, accurate to at least 0.1% is encouraged. The peak rate value may be set to positive infinity, indicating that it is unknown or unspecified.

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

The minimum policed unit, [m], is an integer measured in bytes. This size includes the application data and all protocol headers at or above the IP level (IP, TCP, UDP, RTP, etc.). It does not include the link-level header size, because these headers will change in size as the packet crosses different portions of the internetwork.

All IP datagrams less than size [m] are treated as being of size m for purposes of resource allocation and policing. The purpose of this parameter is to allow reasonable estimation of the per-packet resources needed to process a flow's packets (maximum packet rate can be computed from the [b] and [m] terms) and to reasonably bound the bandwidth overhead consumed by the flow's link-level packet headers. The maximum bandwidth overhead consumed by link-level headers when carrying a flow's packets is bounded by the ratio of the link-level header size to [m]. Without the [m] term, it would be necessary to compute this bandwidth overhead assuming that every flow was always sending minimum-sized packets, which is unacceptable.

The maximum packet size, [M], is the biggest packet that will conform to the traffic specification; it is also measured in bytes. Any packets of larger size sent into the network may not receive QoS-controlled service, since they are considered to not meet the traffic specification.

Both [m] and [M] must be positive, and [m] must be less then or equal to [M].

The XDR description of this parameter is:

```
struct {
  float r;
  float b;
  float p;
  unsigned m;
  unsigned M;
} TOKEN_BUCKET_TSPEC;
```

For the fields [r] and [b] only valid non-negative floating point numbers are allowed. Negative numbers (including "negative zero), infinities, and NAN's are not allowed.

For the field [p], only valid non-negative floating point numbers or positive infinity are allowed. Negative numbers (including "negative zero), negative infinities, 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 parameter values meet these requirements before using the values in floating-point computations, in order to avoid unexpected exceptions or traps.

4. Security Considerations

Implementation of the characterization parameters described in this memo creates no known new avenues for malicious attack on the network infrastructure. Implementation of these characterization parameters does, of necessity, reveal some additional information about a network's performance, which in extremely rare circumstances could be viewed as a security matter by the network provider.

5. References

[RFC 2005] Braden, R., Ed., et. al., "Resource Reservation Protocol (RSVP) - Version 1 Functional Specification", RFC 2205, September 1997.

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

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

[RFC 2212] Shenker, S., Partridge, C., and R. Guerin "Specification of the Guaranteed Quality of Service", RFC 2212, September 1997.

[RFC 2211] Wroclawski, J., "Specification of the Controlled Load Quality of Service", RFC 2211, September 1997.

[RFC 1832] Srinivansan, R., "XDR: External Data Representation Standard", RFC 1832, August 1995.

Authors' Addresses

Scott Shenker
Xerox PARC
3333 Coyote Hill Road
Palo Alto, CA 94304-1314

Phone: 415-812-4840
Fax: 415-812-4471
EMail: shenker@parc.xerox.com

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

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