Network Working Group
Request for Comments: 2430
Category: Informational

T. Li
Juniper Networks
Y. Rekhter
Cisco Systems

October 1998

A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)

Status of this Memo

Copyright Notice

1.0 Abstract

This document describes the Provider Architecture for Differentiated Services and Traffic Engineering (PASTE) for Internet Service Providers (ISPs). Providing differentiated services in ISPs is a challenge because the scaling problems presented by the sheer number of flows present in large ISPs makes the cost of maintaining per-flow state unacceptable. Coupled with this, large ISPs need the ability to perform traffic engineering by directing aggregated flows of traffic along specific paths.

PASTE addresses these issues by using Multiprotocol Label Switching (MPLS) [1] and the Resource Reservation Protocol (RSVP) [2] to create a scalable traffic management architecture that supports differentiated services. This document assumes that the reader has at least some familiarity with both of these technologies.

2.0 Terminology

In common usage, a packet flow, or a flow, refers to a unidirectional stream of packets, distributed over time. Typically a flow has very fine granularity and reflects a single interchange between hosts, such as a TCP connection. An aggregated flow is a number of flows that share forwarding state and a single resource reservation along a sequence of routers.

One mechanism for supporting aggregated flows is Multiprotocol Label Switching (MPLS). In MPLS, packets are tunneled by wrapping them in a minimal header [3]. Each such header contains a label, that carries both forwarding and resource reservation semantics. MPLS defines mechanisms to install label-based forwarding information along a series of Label Switching Routers (LSRs) to construct a Label Switched Path (LSP). LSPs can also be associated with resource reservation information.

One protocol for constructing such LSPs is the Resource Reservation Protocol (RSVP) [4]. When used with the Explicit Route Object (ERO) [5], RSVP can be used to construct an LSP along an explicit route [6].

To support differentiated services, packets are divided into separate traffic classes. For conceptual purposes, we will discuss three different traffic classes: Best Effort, Priority, and Network Control. The exact number of subdivisions within each class is to be defined.

Network Control traffic primarily consists of routing protocols and network management traffic. If Network Control traffic is dropped, routing protocols can fail or flap, resulting in network instability. Thus, Network Control must have very low drop preference. However, Network Control traffic is generally insensitive to moderate delays and requires a relatively small amount of bandwidth. A small bandwidth guarantee is sufficient to insure that Network Control traffic operates correctly.

Priority traffic is likely to come in many flavors, depending on the application. Particular flows may require bandwidth guarantees, jitter guarantees, or upper bounds on delay. For the purposes of this memo, we will not distinguish the subdivisions of priority traffic. All priority traffic is assumed to have an explicit resource reservation.

Currently, the vast majority of traffic in ISPs is Best Effort traffic. This traffic is, for the most part, delay insensitive and reasonably adaptive to congestion.

When flows are aggregated according to their traffic class and then the aggregated flow is placed inside a LSP, we call the result a traffic trunk, or simply a trunk. The traffic class of a packet is orthogonal to the LSP that it is on, so many different trunks, each with its own traffic class, may share an LSP if they have different traffic classes.

3.0 Introduction

The next generation of the Internet presents special challenges that must be addressed by a single, coordinated architecture. While this architecture allows for distinction between ISPs, it also defines a framework within which ISPs may provide end-to-end differentiated services in a coordinated and reliable fashion. With such an architecture, an ISP would be able to craft common agreements for the handling of differentiated services in a consistent fashion, facilitating end-to-end differentiated services via a composition of these agreements. Thus, the goal of this document is to describe an architecture for providing differentiated services within the ISPs of the Internet, while including support for other forthcoming needs such as traffic engineering. While this document addresses the needs of the ISPs, its applicability is not limited to the ISPs. The same architecture could be used in any large, multiprovider catenet needing differentiated services.

This document only discusses unicast services. Extensions to the architecture to support multicast are a subject for future research.

One of the primary considerations in any ISP architecture is scalability. Solutions that have state growth proportional to the size of the Internet result in growth rates exceeding Moore’s law, making such solutions intractable in the long term. Thus, solutions that use mechanisms with very limited growth rates are strongly preferred.

Discussions of differentiated services to date have frequently resulted in solutions that require per-flow state or per-flow queuing. As the number of flows in an ISP within the "default-free zone of the Internet" scales with the size of the Internet, the growth rate is difficult to support and argues strongly for a solution with lower state requirements. Simultaneously, supporting differentiated services is a significant benefit to most ISPs. Such support would allow providers to offer special services such as priority for bandwidth for mission critical services for users willing to pay a service premium. Customers would contract with ISPs for these services under Service Level Agreements (SLAs). Such an agreement may specify the traffic volume, how the traffic is handled, either in an absolute or relative manner, and the compensation that the ISP receives.

Differentiated services are likely to be deployed across a single ISP to support applications such as a single enterprise’s Virtual Private Network (VPN). However, this is only the first wave of service implementation. Closely following this will be the need for differentiated services to support extranets, enterprise VPNs that

span ISPs, or industry interconnection networks such as the ANX [7]. Because such applications span enterprises and thus span ISPs, there is a clear need for inter-domain SLAs. This document discusses the technical architecture that would allow the creation of such inter-domain SLAs.

Another important consideration in this architecture is the advent of traffic engineering within ISPs. Traffic engineering is the ability to move trunks away from the path selected by the ISP's IGP and onto a different path. This allows an ISP to route traffic around known points of congestion in its network, thereby making more efficient use of the available bandwidth. In turn, this makes the ISP more competitive within its market by allowing the ISP to pass lower costs and better service on to its customers.

Finally, the need to provide end-to-end differentiated services implies that the architecture must support consistent inter-provider differentiated services. Most flows in the Internet today traverse multiple ISPs, making a consistent description and treatment of priority flows across ISPs a necessity.

4.0 Components of the Architecture

The Differentiated Services Backbone architecture is the integration of several different mechanisms that, when used in a coordinated way, achieve the goals outlined above. This section describes each of the mechanisms used in some detail. Subsequent sections will then detail the interoperation of these mechanisms.

4.1 Traffic classes

As described above, packets may fall into a variety of different traffic classes. For ISP operations, it is essential that packets be accurately classified before entering the ISP and that it is very easy for an ISP device to determine the traffic class for a particular packet.

The traffic class of MPLS packets can be encoded in the three bits reserved for CoS within the MPLS label header. In addition, traffic classes for IPv4 packets can be classified via the IPv4 ToS byte, possibly within the three precedence bits within that byte. Note that the consistent interpretation of the traffic class, regardless of the bits used to indicate this class, is an important feature of PASTE.

In this architecture it is not overly important to control which packets entering the ISP have a particular traffic class. From the ISP's perspective, each Priority packet should involve some economic premium for delivery. As a result the ISP need not pass judgment as to the appropriateness of the traffic class for the application.

It is important that any Network Control traffic entering an ISP be handled carefully. The contents of such traffic must also be carefully authenticated. Currently, there is no need for traffic generated external to a domain to transit a border router of the ISP.

4.2 Trunks

As described above, traffic of a single traffic class that is aggregated into a single LSP is called a traffic trunk, or simply a trunk. Trunks are essential to the architecture because they allow the overhead in the infrastructure to be decoupled from the size of the network and the amount of traffic in the network. Instead, as the traffic scales up, the amount of traffic in the trunks increases; not the number of trunks.

The number of trunks within a given topology has a worst case of one trunk per traffic class from each entry router to each exit router. If there are N routers in the topology and C classes of service, this would be (N * (N-1) * C) trunks. Fortunately, instantiating this many trunks is not always necessary.

Trunks with a single exit point which share a common internal path can be merged to form a single sink tree. The computation necessary to determine if two trunks can be merged is straightforward. If, when a trunk is being established, it intersects an existing trunk with the same traffic class and the same remaining explicit route, the new trunk can be spliced into the existing trunk at the point of intersection. The splice itself is straightforward: both incoming trunks will perform a standard label switching operation, but will result in the same outbound label. Since each sink tree created this way touches each router at most once and there is one sink tree per exit router, the result is N * C sink trees.

The number of trunks or sink trees can also be reduced if multiple trunks or sink trees for different classes follow the same path. This works because the traffic class of a trunk or sink tree is orthogonal to the path defined by its LSP. Thus, two trunks with different traffic classes can share a label for any part of the topology that is shared and ends in the exit router. Thus, the entire topology can be overlaid with N trunks.

Further, if Best Effort trunks and individual Best Effort flows are treated identically, there is no need to instantiate any Best Effort trunk that would follow the IGP computed path. This is because the packets can be directly forwarded without an LSP. However, traffic engineering may require Best Effort trunks to be treated differently from the individual Best Effort flows, thus requiring the instantiation of LSPs for Best Effort trunks. Note that Priority trunks must be instantiated because end-to-end RSVP packets to support the aggregated Priority flows must be tunneled.

Trunks can also be aggregated with other trunks by adding a new label to the stack of labels for each trunk, effectively bundling the trunks into a single tunnel. For the purposes of this document, this is also considered a trunk, or if we need to be specific, this will be called an aggregated trunk. Two trunks can be aggregated if they share a portion of their path. There is no requirement on the exact length of the common portion of the path, and thus the exact requirements for forming an aggregated trunk are beyond the scope of this document. Note that traffic class (i.e., QoS indication) is propagated when an additional label is added to a trunk, so trunks of different classes may be aggregated.

Trunks can be terminated at any point, resulting in a deaggregation of traffic. The obvious consequence is that there needs to be sufficient switching capacity at the point of deaggregation to deal with the resultant traffic.

High reliability for a trunk can be provided through the use of one or more backup trunks. Backup trunks can be initiated either by the same router that would initiate the primary trunk or by another backup router. The status of the primary trunk can be ascertained by the router that initiated the backup trunk (note that this may be either the same or a different router as the router that initiated the primary trunk) through out of band information, such as the IGP. If a backup trunk is established and the primary trunk returns to service, the backup trunk can be deactivated and the primary trunk used instead.

4.3 RSVP

Originally RSVP was designed as a protocol to install state associated with resource reservations for individual flows originated/destined to hosts, where path was determined by destination-based routing. Quoting directly from the RSVP specifications, "The RSVP protocol is used by a host, on behalf of an application data stream, to request a specific quality of service (QoS) from the network for particular data streams or flows" [RFC2205].

The usage of RSVP in PASTE is quite different from the usage of RSVP as it was originally envisioned by its designers. The first difference is that RSVP is used in PASTE to install state that applies to a collection of flows that all share a common path and common pool of reserved resources. The second difference is that RSVP is used in PASTE to install state related to forwarding, including label switching information, in addition to resource reservations. The third difference is that the path that this state is installed along is no longer constrained by the destination-based routing.

The key factor that makes RSVP suitable for PASTE is the set of mechanisms provided by RSVP. Quoting from the RSVP specifications, "RSVP protocol mechanisms provide a general facility for creating and maintaining distributed reservation state across a mesh of multicast or unicast delivery paths." Moreover, RSVP provides a straightforward extensibility mechanism by allowing for the creation of new RSVP Objects. This flexibility allows us to also use the mechanisms provided by RSVP to create and maintain distributed state for information other than pure resource reservation, as well as allowing the creation of forwarding state in conjunction with resource reservation state.

The original RSVP design, in which "RSVP itself transfers and manipulates QoS control parameters as opaque data, passing them to the appropriate traffic control modules for interpretation" can thus be extended to include explicit route parameters and label binding parameters. Just as with QoS parameters, RSVP can transfer and manipulate explicit route parameters and label binding parameters as opaque data, passing explicit route parameters to the appropriate forwarding module, and label parameters to the appropriate MPLS module.

Moreover, an RSVP session in PASTE is not constrained to be only between a pair of hosts, but is also used between pairs of routers that act as the originator and the terminator of a traffic trunk.

Using RSVP in PASTE helps consolidate procedures for several tasks: (a) procedures for establishing forwarding along an explicit route, (b) procedures for establishing a label switched path, and (c) RSVP’s existing procedures for resource reservation. In addition, these functions can be cleanly combined in any manner. The main advantage of this consolidation comes from an observation that the above three tasks are not independent, but inter-related. Any alternative that accomplished each of these functions via independent sets of procedures, would require additional coordination between functions, adding more complexity to the system.

4.4 Traffic Engineering

The purpose of traffic engineering is to give the ISP precise control over the flow of traffic within its network. Traffic engineering is necessary because standard IGPs compute the shortest path across the ISP's network based solely on the metric that has been administratively assigned to each link. This computation does not take into account the loading of each link. If the ISP's network is not a full mesh of physical links, the result is that there may not be an obvious way to assign metrics to the existing links such that no congestion will occur given known traffic patterns. Traffic engineering can be viewed as assistance to the routing infrastructure that provides additional information in routing traffic along specific paths, with the end goal of more efficient utilization of networking resources.

Traffic engineering is performed by directing trunks along explicit paths within the ISP's topology. This diverts the traffic away from the shortest path computed by the IGP and presumably onto uncongested links, eventually arriving at the same destination. Specification of the explicit route is done by enumerating an explicit list of the routers in the path. Given this list, traffic engineering trunks can be constructed in a variety of ways. For example, a trunk could be manually configured along the explicit path. This would involve configuring each router along the path with state information for forwarding the particular label. Such techniques are currently used for traffic engineering in some ISPs today.

Alternately, a protocol such as RSVP can be used with an Explicit Route Object (ERO) so that the first router in the path can establish the trunk. The computation of the explicit route is beyond the scope of this document but may include considerations of policy, static and dynamic bandwidth allocation, congestion in the topology and manually configured alternatives.

4.5 Resource reservation

Priority traffic has certain requirements on capacity and traffic handling. To provide differentiated services, the ISP's infrastructure must know of, and support these requirements. The mechanism used to communicate these requirements dynamically is RSVP. The flow specification within RSVP can describe many characteristics of the flow or trunk. An LSR receiving RSVP information about a flow or trunk has the ability to look at this information and either accept or reject the reservation based on its local policy. This policy is likely to include constraints about the traffic handling functions that can be supported by the network and the aggregate capacity that the network is willing to provide for Priority traffic.

4.6 Inter-Provider SLAs (IPSs)

Trunks that span multiple ISPs are likely to be based on legal agreements and some other external considerations. As a result, one of the common functions that we would expect to see in this type of architecture is a bilateral agreement between ISPs to support differentiated services. In addition to the obvious compensation, this agreement is likely to spell out the acceptable traffic handling policies and capacities to be used by both parties.

Documents similar to this exist today on behalf of Best Effort traffic and are known as peering agreements. Extending a peering agreement to support differentiated services would effectively create an Inter-Provider SLA (IPS). Such agreements may include the types of differentiated services that one ISP provides to the other ISP, as well as the upper bound on the amount of traffic associated with each such service that the ISP would be willing to accept and carry from the other ISP. Further, an IPS may limit the types of differentiated services and an upper bound on the amount of traffic that may originate from a third party ISP and be passed from one signer of the IPS to the other.

If the expected costs associated with the IPS are not symmetric, the parties may agree that one ISP will provide the other ISP with appropriate compensation. Such costs may be due to inequality of traffic exchange, costs in delivering the exchanged traffic, or the overhead involved in supporting the protocols exchanged between the two ISPs.

Note that the PASTE architecture provides a technical basis to establish IPSs, while the procedures necessary to create such IPSs are outside the scope of PASTE.

4.7 Traffic shaping and policing

To help support IPSs, special facilities must be available at the interconnect between ISPs. These mechanisms are necessary to insure that the network transmitting a trunk of Priority traffic does so within the agreed traffic characterization and capacity. A simplistic example of such a mechanism might be a token bucket system, implemented on a per-trunk basis. Similarly, there need to be mechanisms to insure, on a per trunk basis, that an ISP receiving a trunk receives only the traffic that is in compliance with the agreement between ISPs.

4.8 Multilateral IPSs

Trunks may span multiple ISPs. As a result, establishing a particular trunk may require more than two ISPs. The result would be a multilateral IPS. This type of agreement is unusual with respect to existing Internet business practices in that it requires multiple participating parties for a useful result. This is also challenging because without a commonly accepted service level definition, there will need to be a multilateral definition, and this definition may not be compatible used in IPSs between the same parties.

Because this new type of agreement may be a difficulty, it may in some cases be simpler for certain ISPs to establish aggregated trunks through other ISPs and then contract with customers to aggregate their trunks. In this way, trunks can span multiple ISPs without requiring multilateral IPSs.

Either or both of these two alternatives are possible and acceptable within this architecture, and the choice is left for the the participants to make on a case-by-case basis.

5.0 The Provider Architecture for differentiated Services and Traffic Engineering (PASTE)

The Provider Architecture for differentiated Services and Traffic Engineering (PASTE) is based on the usage of MPLS and RSVP as mechanisms to establish differentiated service connections across ISPs. This is done in a scalable way by aggregating differentiated flows into traffic class specific MPLS tunnels, also known as traffic trunks.

Such trunks can be given an explicit route by an ISP to define the placement of the trunk within the ISP’s infrastructure, allowing the ISP to traffic engineer its own network. Trunks can also be aggregated and merged, which helps the scalability of the architecture by minimizing the number of individual trunks that intermediate systems must support.

Special traffic handling operations, such as specific queuing algorithms or drop computations, can be supported by a network on a per-trunk basis, allowing these services to scale with the number of trunks in the network.

Agreements for handling of trunks between ISPs require both legal documentation and conformance mechanisms on both sides of the agreement. As a trunk is unidirectional, it is sufficient for the transmitter to monitor and shape outbound traffic, while the receiver polices the traffic profile.

Trunks can either be aggregated across other ISPs or can be the subject of a multilateral agreement for the carriage of the trunk. RSVP information about individual flows is tunneled in the trunk to provide an end-to-end reservation. To insure that the return RSVP traffic is handled properly, each trunk must also have another tunnel running in the opposite direction. Note that the reverse tunnel may be a different trunk or it may be an independent tunnel terminating at the same routers as the trunk. Routing symmetry between a trunk and its return is not assumed.

RSVP already contains the ability to do local path repair. In the event of a trunk failure, this capability, along with the ability to specify abstractions in the ERO, allows RSVP to re-establish the trunk in many failure scenarios.

6.0 Traffic flow in the PASTE architecture

As an example of the operation of this architecture, we consider an example of a single differentiated flow. Suppose that a user wishes to make a telephone call using a Voice over IP service. While this call is full duplex, we can consider the data flow in each direction in a half duplex fashion because the architecture operates symmetrically.

Suppose that the data packets for this voice call are created at a node S and need to traverse to node D. Because this is a voice call, the data packets are encoded as Priority packets. If there is more granularity within the traffic classes, these packets might be encoded as wanting low jitter and having low drop preference. Initially this is encoded into the precedence bits of the IPv4 ToS byte.

6.1 Propagation of RSVP messages

To establish the flow to node D, node S first generates an RSVP PATH message which describes the flow in more detail. For example, the flow might require 3kbps of bandwidth, be insensitive to jitter of less than 50ms, and require a delay of less than 200ms. This message is passed through node S’s local network and eventually appears in node S’s ISP. Suppose that this is ISP F.

ISP F has considerable latitude in its options at this point. The requirement on F is to place the flow into a trunk before it exits F’s infrastructure. One thing that F might do is to perform the admission control function at the first hop router. At this point, F would determine if it had the capacity and capability of carrying the flow across its own infrastructure to an exit router E. If the admission control decision is negative, the first hop router can

inform node S using RSVP. Alternately, it can propagate the RSVP PATH message along the path to exit router E. This is simply normal operation of RSVP on a differentiated flow.

At exit router E, there is a trunk that ISP F maintains that transits ISP X, Y, and Z and terminates in ISP L. Based on BGP path information or on out of band information, Node D is known to be a customer of ISP L. Exit router E matches the flow requirements in the RSVP PATH message to the characteristics (e.g., remaining capacity) of the trunk to ISP L. Assuming that the requirements are compatible, it then notes that the flow should be aggregated into the trunk.

To insure that the flow reservation happens end to end, the RSVP PATH message is then encapsulated into the trunk itself, where it is transmitted to ISP L. It eventually reaches the end of the trunk, where it is decapsulated by router U. PATH messages are then propagated all the way to the ultimate destination D.

Note that the end-to-end RSVP RESV messages must be carefully handled by router U. The RESV messages from router U to E must return via a tunnel back to router E.

RSVP is also used by exit router E to initialize and maintain the trunk to ISP L. The RSVP messages for this trunk are not placed within the trunk itself but the end-to-end RSVP messages are. The existence of multiple overlapping RSVP sessions in PASTE is straightforward, but requires explicit enumeration when discussing particular RSVP sessions.

6.2 Propagation of user data

Data packets created by S flow through ISP F's network following the flow reservation and eventually make it to router E. At that point, they are given an MPLS label and placed in the trunk. Normal MPLS switching will propagate this packet across ISP X's network. Note that the same traffic class still applies because the class encoding is propagated from the precedence bits of the IPv4 header to the CoS bits in the MPLS label. As the packet exits ISP X's network, it can be aggregated into another trunk for the express purpose of tranisiting ISP Y.

Again, label switching is used to bring the packet across ISP Y's network and then the aggregated trunk terminates at a router in ISP Z's network. This router deaggregates the trunk, and forwards the resulting trunk towards ISP L. This trunk transits ISP Z and terminates in ISP L at router U. At this point, the data packets are removed from the trunk and forwarded along the path computed by RSVP.

6.3 Trunk establishment and maintenance

In this example, there are two trunks in use. One trunk runs from ISP F, through ISPs X, Y and Z, and then terminates in ISP L. The other aggregated trunk begins in ISP X, transits ISP Y and terminates in ISP Z.

The first trunk may be established based on a multilateral agreement between ISPs F, X, Z and L. Note that ISP Y is not part of this multilateral agreement, and ISP X is contractually responsible for providing carriage of the trunk into ISP Z. Also per this agreement, the tunnel is maintained by ISP F and is initialized and maintained through the use of RSVP and an explicit route object that lists ISP's X, Z, and L. Within this explicit route, ISP X and ISP L are given as strict hops, thus constraining the path so that there may not be other ISPs intervening between the pair of ISPs F and X and the pair Z and L. However, no constraint is placed on the path between ISPs X and Z. Further, there is no constraint placed on which router terminates the trunk within L's infrastructure.

Normally this trunk is maintained by one of ISP F's routers adjacent to ISP X. For robustness, ISP F has a second router adjacent to ISP X, and that provides a backup trunk.

The second trunk may be established by a bilateral agreement between ISP X and Y. ISP Z is not involved. The second trunk is constrained so that it terminates on the last hop router within Y's infrastructure. This tunnel is initialized and maintained through the use of RSVP and an explicit route that lists the last hop router within ISP Y's infrastructure. In order to provide redundancy in the case of the failure of the last hop router, there are multiple explicit routes configured into ISP X's routers. These routers can select one working explicit route from their configured list. Further, in order to provide redundancy against the failure of X's primary router, X provides a backup router with a backup trunk.

6.4 Robustness

Note that in this example, there are no single points of failure once the traffic is within ISP F's network. Each trunk has a backup trunk to protect against the failure of the primary trunk. To protect against the failure of any particular router, each trunk can be configured with multiple explicit route objects that terminate at one of several acceptable routers.

7.0 Security Considerations

Because Priority traffic intrinsically has more 'value' than Best Effort traffic, the ability to inject Priority traffic into a network must be carefully controlled. Further, signaling concerning Priority traffic has to be authenticated because it is likely that the signaling information will result in specific accounting and eventually billing for the Priority services. ISPs are cautioned to insure that the Priority traffic that they accept is in fact from a known previous hop. Note that this is a simple requirement to fulfill at private peerings, but it is much more difficult at public interconnects. For this reason, exchanging Priority traffic at public interconnects should be done with great care.

RSVP traffic needs to be authenticated. This can possibly be done through the use of the Integrity Object.

8.0 Conclusion

The Provider Architecture for differentiated Services and Traffic Engineering (PASTE) provides a robust, scalable means of deploying differentiated services in the Internet. It provides scalability by aggregating flows into class specific MPLS tunnels. These tunnels, also called trunks, can in turn be aggregated, thus leading to a hierarchical aggregation of traffic.

Trunk establishment and maintenance is done with RSVP, taking advantage of existing work in differentiated services. Explicit routes within the RSVP signaling structure allow providers to perform traffic engineering by placing trunks on particular links in their network.

The result is an architecture that is sufficient to scale to meet ISP needs and can provide differentiated services in the large, support traffic engineering, and continue to grow with the Internet.

8.1 Acknowledgments

Inspiration and comments about this document came from Noel Chiappa, Der-Hwa Gan, Robert Elz, Lisa Bourgeault, and Paul Ferguson.

9.0 References

[1] Rosen, E., Viswanathan, A., and R. Callon, "A Proposed Architecture for MPLS", Work in Progress.

[2] Braden, R., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[3] Rosen, E., Rekhter, Y., Tappan, D., Farinacci, D., Fedorkow,, G., Li, T., and A. Conta, "MPLS Label Stack Encoding", Work in Progress.

[4] Davie, B., Rekhter, Y., Rosen, E., Viswanathan, A., and V. Srinivasan, "Use of Label Switching With RSVP", Work in Progress.

[5] Gan, D.-H., Guerin, R., Kamat, S., Li, T., and E. Rosen, "Setting up Reservations on Explicit Paths using RSVP", Work in Progress.

[6] Davie, B., Li, T., Rosen, E., and Y. Rekhter, "Explicit Route Support in MPLS", Work in Progress.

[7] http://www.anxo.com/

10.0 Authors' Addresses

Tony Li
Juniper Networks, Inc.
385 Ravendale Dr.
Mountain View, CA 94043

Phone: +1 650 526 8006
Fax: +1 650 526 8001
EMail: tli@juniper.net

Yakov Rekhter
cisco Systems, Inc.
170 W. Tasman Dr.
San Jose, CA 95134

EMail: yakov@cisco.com

11. Full Copyright Statement