BGRP: A Tree-Based Aggregation Protocol for Inter-Domain Reservations

Ping Pan
Bell Laboratories
101 Crawfords Corner Rd.
Holmdel, NJ 07733
pingpan@bell-labs.com

Ellen L. Hahne
Bell Laboratories
600 Mountain Ave.
Murray Hill, NJ 07974
hahne@bell-labs.com

Henning Schulzrinne
Dept. of Computer Science
Columbia University
New York, NY 10027
hgs@cs.columbia.edu

December 6, 1999

Abstract

Resource reservation needs to accommodate the rapidly growing size and increasing service diversity of the Internet. Recently, hierarchical architectures have been proposed that provide domain-level reservation. However, it is not clear that these proposals can set up and maintain reservations in an efficient and scalable fashion.

In this paper, we describe a distributed architecture for inter-domain aggregated resource reservation for unicast traffic. We also present an associated protocol, called the Border Gateway Reservation Protocol (BGRP), that scales well, in terms of message processing load, state storage and bandwidth. Each stub or transit domain may use its own intra-domain resource reservation protocol. BGRP builds a sink tree for each of the stub domains. Each sink tree aggregates bandwidth reservations from all data sources in the network. Since backbone routers only maintain the sink tree information, the total number of reservations at each router scales linearly with the number of domains in the Internet. (Even aggregated versions of the current protocol RSVP have an overhead that grows like N^2.)

BGRP relies on Differentiated Services for data forwarding. As a result, the number of packet classifier entries is extremely small. To reduce the protocol message traffic, routers may reserve domain bandwidth beyond the current load, so that sources can join or leave the tree or change their reservation without having to send messages all the way to the tree root for every such change. We use "soft state" to maintain reservations. In contrast to RSVP, refresh messages are delivered reliably, allowing us to reduce the refresh frequency.

Columbia University Computer Science Technical Report No. CUCS-029-99

1 Introduction

Resource reservation was originally defined to support end-to-end QoS guarantees for a range of QoS-sensitive applications, including multimedia-on-demand and teleconferencing. Recently, Internet Service Providers (ISPs) have started to use the same reservation mechanisms to provide customer-level Virtual Private Networks (VPNs) and to allocate network resources between Differentiated Service classes. We believe that the applications of resource reservation and the demand for it will continue to grow, and that reserved QoS will eventually come to be seen as an indispensable feature of Internet service. At the present time, there are three challenges to the widespread use of reserved QoS: reservation protocol scalability, packet forwarding scalability, and inter-domain management.

Reservation protocol scalability: First, resource reservation schemes must scale well with the rapidly growing size of the Internet. A router may be able to handle tens of thousands of simultaneous reservations [1], but not hundreds of thousands, and certainly not millions. Today's traffic volume is bad enough: as we will show in Table 1 of Sec. 5.1 below, we have measured hundreds of thousands to millions of flows at the MAE-West network access point; if many of these flows were to request resource reservations, the protocol overhead would swamp the router. But projected future traffic growth is an even more serious problem. The overhead of the current protocol RSVP [2, 3] grows like N^2, where N is the number of Internet end hosts. Fig 1 shows the growth of N over the last six years, from 2 million to 60 million. This means that N^2 grew from $4 \cdot 10^{12}$ to $4 \cdot 10^{15}$ during that time! With no end in sight, N^2 is growing much faster than improvements in processing speeds or memory sizes. Therefore, we will have to find a reservation scheme that scales better than conventional RSVP. In this paper we will propose a protocol, called the Border Gateway Reservation Protocol (BGRP), that fixes this scaling problem in two ways. First, BGRP overhead scales *linearly* with the size of the Internet; i.e., N^2 is reduced to N. Second, BGRP uses "a smaller N". The overhead of the basic BGRP protocol is proportional to the number of Internet carrier domains (also called Autonomous Systems (AS)), while an enhanced version of BGRP has overhead proportional to the number of IP networks (i.e., the number of announced IP address prefixes).

Data forwarding scalability: In addition to the overhead of the protocol that *reserves* QoS, another scaling issue is the overhead of the packet classifiers, enqueuers and schedulers that *enforce* the reserved QoS[4]. The RSVP protocol was designed to work within the Integrated Services (IntServ) architecture [5]. IntServ requires backbone routers to classify and schedule packets on a per-flow basis. Hence, the data forwarding overhead with RSVP/IntServ also grows like the square of the number of end hosts. Our proposed BGRP protocol, however, is designed to work within the Differentiated Services (DiffServ) architecture [6]. With DiffServ, all flow-related handling of packets (e.g., classification, policing, shaping) is done at the edges of the network. At the edge, packets are assigned to one of a few dozen QoS classes. The QoS class of a packet is recorded in the DiffServ Byte of its IP header. Backbone routers queue and schedule packets according to their QoS class only; i.e., there is no flow-based packet classification, queueing, or scheduling in the backbone. Hence, the data forwarding operations used with BGRP are quite scalable.

Inter-domain adminstration: In general, Internet flows traverse several different network domains. Each ISP would prefer to manage its own network resources and enforce its own internal traffic engineering policies[7], acting as independently as possible of other carriers. Ideally, a domain should only have to reveal simple delivery commitments to its peering domains. There should be an inter-domain reservation system that uses these delivery commitments to establish a reservation path through multiple domains. Each domain would set up transit reservation flows using its own preferred intra-domain reservation mechanism. BGRP is specifically designed for such an environment. It operates on the

boundaries between ISPs, leaving each ISP free to manage its own domain independently. Once QoS traffic management conforms itself to the technical and business "topology" of the Internet – a loosely coupled collection of competing and mistrustful carriers, barely cooperating through bilateral peer routing arrangements – then *charging* customers for QoS should become practical. We expect that the forging of this final missing link will encourage explosive deployment of QoS reservation mechanisms.

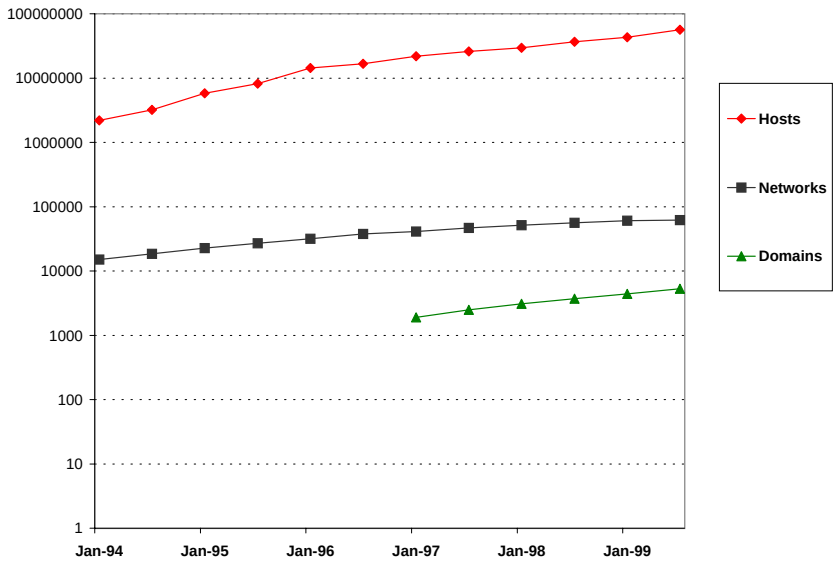


Figure 1: The growth of the Internet from 1994 to 1999[8, 9, 10].

2 Aggregating Reservations

How many simultaneous reservations can a router handle? In a recent study [1], we showed that a low-end router can set up 900 new RSVP reservations or refresh up to 1600 reservations per second, allowing it to sustain about 45,000 flows. To handle that many reservations, the router has to suspend routing computation and packet forwarding, due to hardware and CPU constraints. While backbone routers have more CPU power than the low-end router used for the measurements, other results [11] indicate that frequent routing computation due to route instability may already tax the CPU. Thus we, along with some other RSVP developers we have consulted, believe that in many networking environments, routers do not have enough CPU power to sustain hundreds of thousands of reservations.[1]

[1]Developers from several gigabit router vendors have acknowledged that RSVP processing could consume between 30% to 50% of router CPU cycles.

If we assume a router can handle tens of thousands of reservations, but not hundreds of thousands, is this enough? There are several ways to go about answering these questions. In this section, we will give some ballpark estimates, based mostly on the data in Fig. 1. Much later, when we evaluate the performace of BGRP in Section 5.1, we will give more precise estimates, based on our statistical analyses of Internet packet traces.

Conventional RSVP reserves resources for each application-to-application flow. For simplicity, let us count host-to-host flows instead. According to Fig. 1, there are $36 \cdot 10^{14}$ source-host/destination-host pairs! This is enormously more than a router can handle, but of course, not all the host-to-host flows will be active at once, and not all active flows will pass through a given router or a given link. So let us try another upper bound on host-to-host flows that is somewhat more realistic: how many reserved flows of non-zero bandwidth can fit on a link of finite bandwidth? Let us assume that 16 kb/s (enough, say, for a good quality packet voice call) is the finest granularity of reservation. Then an OC-192 (10 Gb/s) link might be called upon to support up to 600,000 reservations. This suggests that we may not be able to afford to reserve for individual host-to-host flows. Some aggregation of reservations is needed.

How might this aggregation be done? There are several ways to aggregate based on "regions" of IP addresses. A "region" can be defined at various granularities; e.g., a region could be one host, or one network, or one AS. The simplest option is to aggregate for each source-region/destination-region pair; i.e., on a given link, aggregate the reservations for all flows *from* a given source region *to* a given destination region. Alternatively, we could aggregate for each source region, i.e., on a given link, aggregate the reservations for all flows *from* a given source region *to* anywhere. Or we could aggregate for each destination region, i.e., on a given link, aggregate the reservations for all flows *from* anywhere *to* a given destination region.

Which of these aggregation options might be adequate in reducing the number of simultaneous reservations to a level that a router can handle? (There is no point in aggregating more than necessary, since this will make the protocol needlessly complex.) According to Fig. 1, there are approximately 60,000,000 hosts, 60,000 networks, and 6,000 AS in the Internet today. That makes $36 \cdot 10^{14}$ host pairs, $36 \cdot 10^{8}$ network pairs, and $36 \cdot 10^{6}$ AS pairs. Comparing these six numbers to our estimate of router capacity determines our target degree of aggregation: one aggregate reservation for each region, where the size of a region is somewhere between one network and one AS. Furthermore, as we shall explain in the next paragraph, we prefer to produce one aggregate reservation for each *destination* region, rather than one per source region, because destination-based aggregates have a convenient tree structure.

In today's Internet, all unicast routing algorithms, including OSPF, RIP and BGP, define sink-based trees, one for each destination. In networks where the shortest paths (or hop-counts) are unique, the principle of optimality guarantees that the shortest paths *to* any *destination* form a tree; the shortest paths *from* any *source* are also guaranteed to form a tree. If there are multiple shortest-length paths, however, the existence of trees depends on the tie-breaking rules in the routing algorithm.

BGP establishes "virtual edges" by using reachability as a definition for the existence of a link in the graph. When there are multiple equal paths, it breaks ties in a way that guarantees sink trees, but not necessarily source trees. Current BGP practice dictates that the router forward all packets to the same destination over only one next-hop router. This practice guarantees that routing paths form sink trees.

Since reservations are made along routes chosen by the routing algorithms, it is natural to aggregate these reservations along the sink trees formed by routing. Hence sink-tree aggregation is the basis of the BGRP protocol that we will present in Sec. 3.

2.1 Paper Outline

This paper is organized as follows. In Section 3, we propose a simple approach for scalable reservation aggregation, and we present the BGRP protocol that achieves it. The basic idea is to build a sink tree for each of the destination domains. Each sink tree aggregates reservations for flows from all data sources in

the network to that destination domain. Section 4 describes various ways to improve the performance of the basic BGRP protocol. The scaling benefits of our approach are evaluated in Sec. 5. We brief discuss some of the recent attempts on scaling reservation in Section 6. We summarize our investigation and describe future work in Section 7.

3 The Border Gateway Reservation Protocol (BGRP)

We propose a distributed architecture in which reservations are aggregated along sink trees and a new signaling protocol, called the Border Gateway Reservation Protocol (BGRP), that supports this architecture.

3.1 Model and Terminology

As shown in Figure 2, the Internet is composed of a number of *domains*, also known as autonomous systems (AS), that exchange user traffic among each other. Each domain is under a common administration and is identified by a unique AS number. A domain in the Internet can be classified as either a *stub* domain or a *transit* domain. A domain is a stub domain if the path connecting any two routers i and j goes through that domain only if at least one of them is part of that domain; transit domains do not have this restriction. A domain connects to a number of other domains via *border routers*. We assume all border routers use BGP [12] for inter-domain routing.

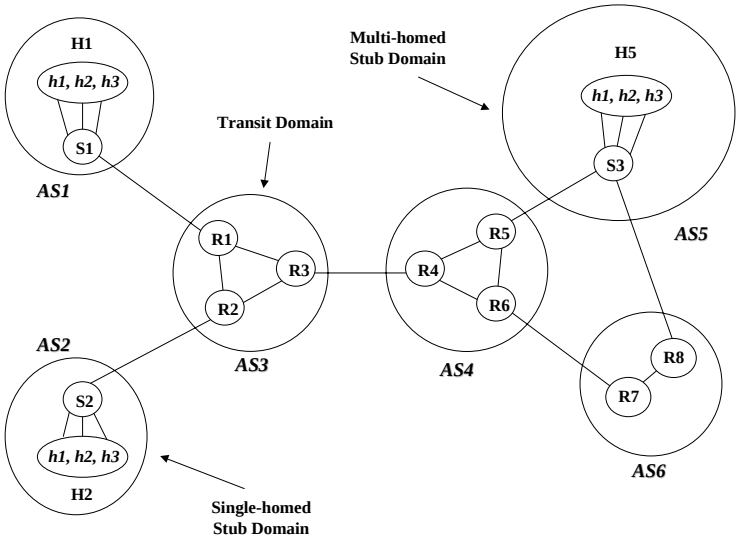


Figure 2: Example of Internet domains. There are two types of stub domains: *single-homed* stub domains connect to the backbone at a single point, *multi-homed* stub domains at several points.

We define $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ as the set of border routers in transit domains, $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ as the set of border routers in stub domains, and $\mathcal{H}_i = \{h_1, h_2, \ldots, h_j\}$ as the set of end hosts in AS$_i$. $\mathcal{H} = \cup\mathcal{H}_i$ comprises all hosts in the network. In this paper, inter-domain reservations originate and terminate at routers in $\mathcal{S}$. We denote the direction of packets traveling from source towards sink as *downstream*, with *upstream* as the opposite direction.

For simplicity, we assume in this paper that all active hosts are in stub domains. (In reality, a transit router R_i could also play the role of a source or sink router for end users in its domain). Moreover, while in reality there are likely to be multiple routers in a domain between border routers, they do not participate in our inter-domain reservation protocol and are thus not shown in our figures.

3.2 Overview of Protocol Operation

The BGRP protocol operates only between border routers, and it interfaces with the BGP routing information base to make routing decisions. We shall use the term *BR-hop* to denote the "virtual hop" between two "adjacent" border routers participating in BGRP. These "adjacent" border routers could be in different domains, i.e., at the two ends of a trans-domain link, or they could be in the same domain. To avoid message delivery delay problems, BGRP protocol messages are reliably delivered. "Soft state", i.e., periodic refresh, protects against events such as link failure.

BGRP defines several control messages: PROBE, GRAFT, REFRESH, ERROR, and TEAR. Reservation sources always initiate PROBE messages to find out about the network resource availability and the exact reservation path. Reservation sinks, upon receiving PROBE messages, return GRAFT messages to set up the appropriate reservations inside the network. PROBE and GRAFT specify the reservation parameters such as traffic class, priority and bandwidth. Reservation sources and sinks transmit PROBE and GRAFT messages only once during the lifetime of a reservation. The border routers periodically exchange REFRESH messages with their peers to make sure reservations are still "alive". In case of reservation failures, the routers send ERROR messages to inform the users. The protocol includes optional TEAR messages that routers can send to actively remove reservations when routes change.

Generally speaking, setting up a reservation consists of finding a path and creating the reservation. One might argue that QoS routing protocols would be sufficient for finding the path, leaving only reservation creation for the reservation protocol to perform. This may be true for the case of intra-domain reservation. But in an inter-domain environment, we believe that determining the reservation path depends as much on ISP policy as on resource availability. Since each ISP only advertises its resource allocation policy to its immediately adjacent peers in the form of bilateral agreement, a resource user may have to actively "probe" the network to determine the edge-to-edge routing path for its reservation.

PROBE messages travel downstream from stub domain border routers. Each PROBE message consist of a reservation request and destination network information. Border routers use BGP routing information and bilateral QoS agreements to forward PROBE messages. PROBE messages collect routing information along the reservation path, similarly to *IP Record Route Option*, but they do not install any reservation state or any routing state in the routers.

To set up the reservation, the destination border router sends a GRAFT message upstream. The GRAFT message uses the previously collected routing information and traverse exactly the reserve direction of the PROBE messages. Routers on the path keep one reservation entry for each sink tree. Among other things, this reservation entry identifies the root of the sink tree. Each reservation root is defined as the combination of the reservation destination information and the ID of the sink border router that processed and terminated the original PROBE message. (The sink border router ID is necessary in case the destination stub domain is multi-homed, because several of that domain's border routers could receive PROBE messages and thus become reservation sinks.) When processing a GRAFT message, each border router interfaces with intra-domain protocols to set up transit reservations within its domain.

Since the majority of the backbone traffic is unicast, and BGRP uses BGP for route selection, we limit ourselves to unicast reservations in this paper and defer multicast support to future work.

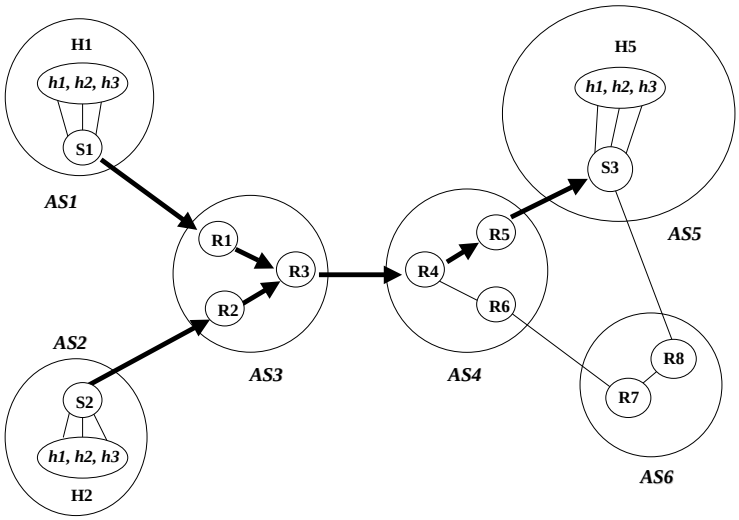


Figure 3: Example of a sink tree rooted at S_3

3.3 Aggregation

Conservative ISPs can allow border routers to simply sum the bandwidth and buffer requirements of the individual BGRP reservations. However, ISPs may take advantage of statistical multiplexing and propagate an aggregated reservation that is somewhat less than the sum of its branches; details are beyond the scope of this paper. To simplify the presentation, we shall assume in the remainder of this paper that bandwidth is the only reserved resource, and that bandwidth reservations are additive.

3.4 PROBE Message: Reservation Route Discovery

Stub domains need to have knowledge of network resource and service availability before setting up inter-domain reservations. BGRP's reservation discovery procedure provides a simple and accurate mechanism for reservation sources to discover a feasible path over multiple domains before the component reservations are actually established along this path.

Figure 3 illustrates how BGRP route discovery works. Suppose H_1 in AS_1 needs to set up a reservation to H_5 in AS_5. All host reservation requests from AS_1 are aggregated at S_1. S_1 sends a PROBE message containing the source ID S_1, the ID of the destination in $\mathcal{H}_5$ (which could be either an application, a host or a subnet), the bandwidth needed, and an empty route record field. In this example, S_1 launches a PROBE at the behest of particular hosts. In a VPN application, however, S_1 could initiate a PROBE message toward S_3 directly, in an effort to set up a virtual "trunk" between their two domains.

When the PROBE message arrives at the border router R_1, the router may consult with domain AS_3's resource database and the bilateral agreement between AS_1 and AS_3 If AS_3 can accept the requested reservation, R_1 will insert its own IP address into the route record field and will forward the PROBE message downstream. Otherwise, R_1 sends an ERROR message back to S_1. The selection of the downstream border router depends on intra-domain traffic engineering requirements [7] and BGP routing policy. In this exam-

ple, R_1 forwards the message to R_3[2]. To prevent loops, each router checks whether the current route record already contains the router's own address.

Let's assume that R_3, R_4 and R_5 all accept the reservation. When the PROBE message arrives at S_3, S_3 determines that the destination in $\mathcal{H}_5$ belongs to its local domain AS_5, so S_3 terminates the probing process. The final route record in the PROBE message is (R_1, R_3, R_4, R_5).

3.5 GRAFT Message: Reservation Creation and Aggregation

Let us continue our example in in Fig. 3. After terminating the PROBE, the destination border router S_3 sends a GRAFT message back toward S_1 to set up the desired reservation along the path. The GRAFT message is source-routed using information gathered in the route record of the PROBE message. The GRAFT message contains the bandwidth requirement $B_{1,3}$, source ID S_1, route record (R_1, R_3, R_4, R_5), sink ID S_3, and a tree ID label L. The tree ID label is always assigned by the sink border router. The label is used to uniquely identify a reservation tree, because there may be multiple reservation trees rooted at a single sink router. A tree ID label can be in the form of a CIDR prefix or the AS number. Assume that this reservation, for bandwidth $B_{1,3}$, is successfully established between S_1 and S_3.

Now suppose that S_3 receives another PROBE, this one regarding a reservation of bandwidth $B_{2,3}$ from S_2 to $\mathcal{H}_5$. S_3 sends back a GRAFT message containing bandwidth requirement $B_{2,3}$, source ID S_2, route record (R_2, R_3, R_4, R_5), sink ID S_3, and the same tree ID label L that it used in the previous GRAFT message. When S_3's GRAFT message reaches R_5, R_5 recognizes this as an increment to the existing tree L, and the reserved bandwidth on the BR-hop between R_5 and S_3 is increased to $BW_{1,3} + BW_{2,3}$. Then R_5 forwards the GRAFT message to R_4 (listed in the route record field), while using the intra-domain reservation protocols of AS_4 to update the internal reservation between R_4 and R_5. (Either R_5 or R_4 could be the initiator of this internal change, depending on the intra-domain reservation protocol.) The reserved bandwidth on the BR-hop between R_4 and R_5 increases to $BW_{1,3} + BW_{2,3}$. Similarly, router R_4 forwards the GRAFT message to R_3 while incrementing the reservation between them. When the GRAFT arrives at R_3, that router creates a new reservation tree branch to R_2 with bandwidth $BW_{2,3}$. The reservation finishes when the GRAFT arrives at S_2.

If a router R_i cannot set up the new reservation, it sends an ERROR message back to the sink to inform it of the failure. Along the way, the ERROR message removes the reservation.

Each border router, e.g., R_3, must maintain the following state information about the reservation tree: the sink tree ID S_3, the tree label L, the next downstream border router R_4, the amount of bandwidth reserved from R_3 to R_4 (viz., $BW_{1,3} + BW_{2,3}$), the adjacent upstream border routers R_1 and R_2, and the amount of bandwidth reserved from each upstream border router to R_3 (viz., $BW_{1,3}$ from R_1 and $BW_{2,3}$ from R_2).

3.6 REFRESH Message: Reservation State Management

We use "soft state" to manage aggregated reservation state among routers. As mentioned above, a router R_i separately stores the reservation information regarding its adjacent upstream and downstream border routers. R_i schedules a REFRESH message to each adjacent router periodically. The message contains the tree ID and the resource reservation information. If R_i has not received the corresponding REFRESH from an upstream or downstream border router within a given refresh interval, R_i will remove the corresponding resource locally. As an option, R_i can actively notify its adjacent routers about the reservation changes.

If a tree leaf S_i wishes to cancel its reservation, it can just stop sending REFRESH messages. This will trigger a chain of reservation timeouts or reservation reductions as successive downstream nodes omit or alter

[2]This is from the BGP NEXT_HOP path attribute.

the REFRESH messages they send downstream. Optionally, the leaf can actively remove the reservation by sending a TEAR message.

If a router experiences a link failure or a routing change affecting its BGRP reservations, the router should stop sending REFRESH messages for these reservations upstream, and the router should reduce the bandwidth declared in its downstream REFRESH messages. Through the periodic refresh mechanism, this will cause a cascade of timeouts in the upstream direction, removing the reservations between the breakpoint and the leaves. It will also trigger a cascade of reservation reductions in the downstream direction, from the breakpoint to the root. Explicit tear-down of reservations is permitted, but it is optional.

3.7 Comparing BGRP with RSVP

BGRPs approach is similar to RSVP, with BGRP's PROBE and GRAFT messages playing similar roles to RSVP's PATH and RESV messages. However, there are some of important differences between the two protocols:

Stateless probing. The primary purpose of RSVP's PATH message is to install routing state at intermediate routers, to guide the RESV message back to the data sender. Routers must therefore keep both sender and destination information. In a network with N nodes, this may require $O(N^2)$ entries. In BGRP, PROBE messages install no state in the routers. Routers only store the reservation information for the $O(N)$ sinks, but no source information.

Reservation aggregation. RSVP can combine reservations in two ways. First, RSVP allows multiple (multicast) receivers to merge their reservations for the same sender (or set of senders) into a single reservation whose size is roughly the *maximum* of the individual reservations. Second, RSVP offers per-source and shared reservation styles; in the latter, multiple (multicast) senders *take turns* sharing a single reservation [13]. BGRP reservation aggregation is different from either of these. BGRP aggregates reservations from different (unicast) senders to the same receiver by *adding* them together.

Bundled refresh. RSVP transmits PATH and RESV messages periodically to refresh each individual reservation, while BGRP bundles all reservation messages during each refresh. Similar enhancements have recently been proposed for RSVP itself [14].

4 BGRP Enhancements

The basic BGRP protocol aggregates reservations into trees, thereby reducing the number of reservations. We will quantify this in Sections 5.1 and 5.2. Reducing the number of reservations obviously shrinks the memory needed to store the control state information. It also reduces the overhead associated with REFRESH messages for all these pieces of control state; refresh costs include CPU processing and link bandwidth. These savings take us much of the way toward our goal. However, BGRP's other control messages, PROBE and GRAFT, also consume processing and bandwidth. We would like to control the volume of these control messages as well and thereby add another dimension of scalability to BGRP. This can be done by making the following enhancements to the protocol.

4.1 Over-reservation, Quantization and Hysteresis

Leaf nodes, in their PROBE message, can request more bandwidth between themselves and the tree root than is currently required. (One can think of this as aggregated advance reservations on behalf of unknown parties.)

Another method is to quantize the amount of reserving bandwidth coarsely, e.g., restrict it to multiples of some quantum Q.

BGP can also employ hysteresis to further reduce the number of messages; e.g., if the bandwidth requested by a leaf node has just jumped from $3Q$ to $4Q$ because its bandwidth requirement has just exceeded $3Q$, then that leaf node should not reduce its request back to $3Q$ until its bandwidth requirement drops below some threshold $T < 3Q$.

These changes can dramatically reduce the volume of control messages, as we will quantify in Sec. 5.4.

4.2 CIDR Labeling and Quiet Grafting

When a sink tree are labelled with CIDR prefix associated with the tree root, the advancing PROBE messages can recognize that the reservation destination belongs to the prefix and thus join the sink tree right away. Also, the tree nodes can over-reserved bandwidth between itself and the tree root. These two modifications enable a new tree operation called *quiet grafting*, whereby a new branch can be grafted onto an existing reservation sink tree without any PROBE or GRAFT messages being passed between the grafting node and the tree root.

Quiet grafting works as follows. A source border router launches a PROBE message, which travels toward the destination. If this PROBE reaches a border router R on the tree it wants to join, and if that tree node R has already over-reserved sufficient bandwidth from itself to the tree root to accomodate the PROBE's bandwidth request, then R handles the reservation directly itself, without propagating the PROBE further downstream. Node R does this by launching a GRAFT message back toward the leaf router, thereby establishing the reservation between the leaf and R. Reserving the requested bandwidth between R and the root is just a matter of R's own internal bookkeeping – R notes that some of the excess bandwidth between itself and the root has now been assigned to the new flow. When the time comes for periodic refreshing of the aggregate reservations between the R and the root, even more bandwidth will be over-reserved, to replenish R's stock.

4.3 Self-Healing (Local Repair)

When a route changes, BGRP has the option of moving the affected reservations to the new route, without demolishing the entire reservation tree and re-creating the tree from scratch. Assume that the reservation tree is labeled with the destination CIDR prefixes, as in Sec. 4.2. When a tree node detects a route change, it can initiate a new PROBE toward the sink. When this PROBE reaches a downstream router on the stable part of the old reservation tree, that router can respond with a GRAFT and thus repair the part of the reservation between the two routers. We call this process *self-healing*.

4.4 Reservation Damping

Labovitz *et al.*[11, 15] have shown that, of three million BGP route changes each day, 99% were pathological and did not reflect real network topological changes. If routers make BGRP reservation changes in response to every route change, there could be a high volume of nearly worthless reservation messages in the network. On the other hand, if the routers do not move a reservation, and the route change turns out to be legitimate and stable, then the data will have lost its reservation. So the trade-off is to determine when to adjust reservations. Here, we propose a damping function for BGRP. The goal of reservation damping is to delay the initiation of the self-healing process of Sec. 4.3 until the changing routes have stabilized. The delay depends on the probability of future instability of the route. Routes that change frequently will be delayed longer. Similar to [16, 17], we propose an exponential function $\tau = (1 + \Delta)^n \cdot T$ for computing the delay

τ between PROBEs sent due to route changes. Δ and T are the parameters to adjust the damping, and n is the number of route changes measured in a time interval.

5 Protocol Scaling Evaluation

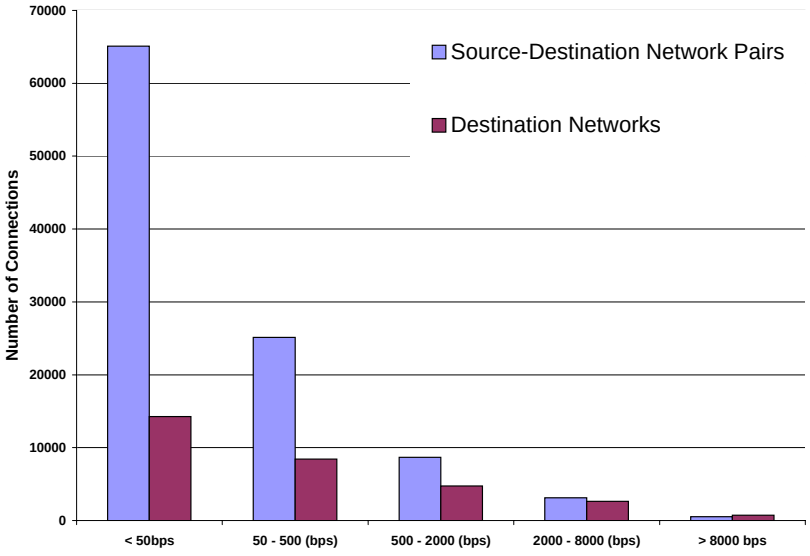
5.1 Estimating Reservation Volume from Packet Traces

We would like to determine how well in practice BGRP will reduce the volume of reservations, as compared with conventional RSVP and its aggregated region-to-region extensions. To that end, we examined a 90-second traffic trace from the MAE-West network access point (NAP).[3] We categorized about 3 million IP packet headers according to their transport-layer port, IP address, IP network prefix, and AS. Table 1 shows the results. Suppose that all traffic desired a reservation of some quality level. If we use conventional RSVP and reserve for each source-destination pair at the application level, then the total number of active reservations would be 339,245. This data strengthens an estimate we made in Sec. 2, viz., that there can be hundreds of thousands of flows on a link, more than a router can handle, and hence some aggregation is necessary. Table 1 also shows that if we use aggregated versions of RSVP to reserve for source-destination pairs of various granularities, we can greatly reduce the number of flows. For example, for AS-to-AS aggregation, the number of flows in the 90-second window was only 20,857. If twenty thousand were as bad as things could get, then a backbone router would be able to handle one RSVP reservation for each source-destination AS pair. However, this number may be artificially low due to the small 90-second window. Let us see what happens if we observe over a month-long window. The last line of Table 1 includes AS counts seen at MAE-West during May 1999.[4] During this month, MAE-West saw 5,001 destination AS; according to Fig. 1, this is essentially the complete AS roster. The number of different source-destination AS pairs viewed that month was enormous: 7,900,362 AS pairs, about a third of the 25 million possible combinations. Thus, unless routers tear down AS-to-AS reservations frequently, there may be too many such AS-level "trunks" to sustain in backbone routers. This data strengthens another estimate we made in Sec. 2, viz., that there can be millions of AS-to-AS flows, more than a router can handle, and hence the point-to-point aggregation style of RSVP is inadequate, even if each "point" is enlarged into a region the size of an AS. What is needed instead is BGRP's style of aggregation, with one reservation per destination AS, or at most one per destination network. The last column of Table 1 shows the gain of BGRP over RSVP, i.e., the ratio of RSVP flows to BGRP flows. The gain is greatest if the aggregation regions are large and if the reservations last a long time. At the AS level of granularity, for a one-month time duration, it is clear that the "N-vs-N^2" problem in theory becomes an "N-vs-N^2" problem in reality. Under these circumstances, BGRP outperforms RSVP by a factor of 1580.

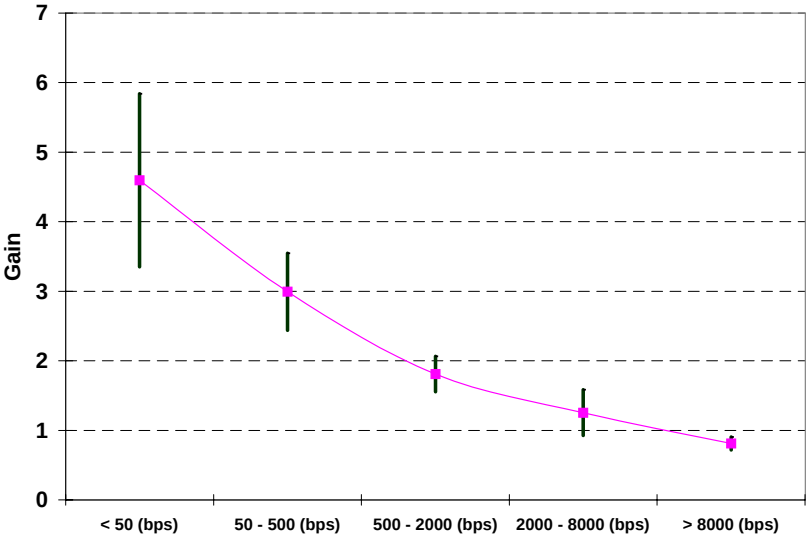
Our assumption above was that every flow would request a reservation. Let us explore this issue further. Conventional wisdom holds that long-lived high-volume real-time applications like packet voice and packet video will want resource reservation. On the other hand, recent IETF proposals [19, 20] have suggested using resource reservation protocols to set up VPN links and to provide service differentiation among users. These reservations can be for scheduling priority as well as bandwidth. The bandwidth of an individual VPN flow or Differentiated Service flow might not be as large as a typical real-time flow, hence there might be even more such reservations on a given backbone link. To see whether BGRP shows the same gains over RSVP for both the "IntServ"-style reserved flows and the "DiffServ"-style reserved flows, we sorted the flows in our packet traces by size. We collected eight packet header traces from MAE West. The traces were collected three hours apart on June 1, 1999 [18]. Each trace comprises three minutes of all traffic at the

[3] The 90-second traces date from June 1, 1999; see [18]. The AS information was collected on June 10, 1999 and analyzed by Sean McCreary of NLANR/CAIDA.

[4] Analysis provided by Sean McCreary of NLANR/CAIDA.



(a) Number of reservations, broken down by bandwidth



(b) Gain for each bandwidth class

Figure 4: Distribution of reservations by bandwidth

Time Interval	Region Granularity	# Source-Dest. Pairs (For RSVP)	# Destinations (For BGRP)	Gain
90 sec.	Application (Session = Address + Port ; Destination = Address + Port + Protocol)	339,245	208,559	1.6
	Host (IP Address)	131,009	40,538	3.2
	Network (CIDR Prefix)	79,786	20,887	3.8
	Domain (AS)	20,857	2,891	7.2
1 month	Domain (AS)	7,900,362	5,001	1,579.8

Table 1: Number of aggregate flows seen in packet traces

NAP and contains about 33 million packet entries. We computed the number of bytes for each destination network address prefix (for BGRP-style reservations), and we also computed the number of bytes for each pair of source and destination prefixes (for RSVP-style reservations). We sorted the data into five categories: fewer than 50 b/s, 50–500 b/s, 500–2000 b/s, 2000–8000 b/s, and greater than 8000 b/s. Figure 4 plots the distribution of reservations by bandwidth, and it also plots the gain (i.e., the ratio of RSVP reservations to BGRP reservations) for each bandwidth class. Most of packets belong to the small-flow category (63.5% for RSVP and 46.2% for BGRP). Only 3621 (3.5%) of the source-destination pairs and 1296 (10.9%) of the destinations have an average bit rate over 2000 b/s. (Interestingly, there are more above-8000 b/s destination-only flows (719) than source-destination flows (516).) To summarize Figure 4, if reservations are made only for high-volume sessions, then such flows are rare enough that RSVP scalability is not an issue, and using BGRP would not significantly reduce the number of such flows anyway. However, with VPN and Differentiated Services, where even the hordes of small flows may require QoS, RSVP scalability could be a problem and BGRP could be quite beneficial.

5.2 Topological Distribution of Demand

We use the simple model in Fig. 5 to compare the scaling properties of BGRP and RSVP. The model depicts a progression of domains along an Internet path, with access networks toward the left and right and backbone networks near the middle of the topology. We define D as the maximum edge-to-edge distance, measured as the number of AS. A "node" n_i in the figure represents an inter-domain traffic exchange point, which can be either a point of presence (POP) or a NAP. (In the real network from which this model is abstracted, each node n_i could actually contain many routers and interconnect many AS.) A "link" l_i in the figure represents an aggregation of all the real links that transport traffic from domain i to domain $i+1$. The model also includes a reverse-directed link from $i+1$ to i, which is not shown in the figure. In addition to the diameter D, our model is characterized by the quantities s_i and d_i: s_i is the number of inter-domain reservation sources coming into n_i, and d_i is the number of reservations sinks reached through n_i, not including those that n_i reaches via l_i.

In this model, the number of RSVP flows (i.e., source-destination pairs) on the uni-directional link l_i is given by

$$L_i^{\text{RSVP}} = \sum_{j=0}^{i} s_j \sum_{k=i+1}^{D} d_k$$

and the number of BGRP flows (i.e., one per destination) on l_i is given by

$$L_i^{\text{BGRP}} = \sum_{k=i+1}^{D} d_k.$$

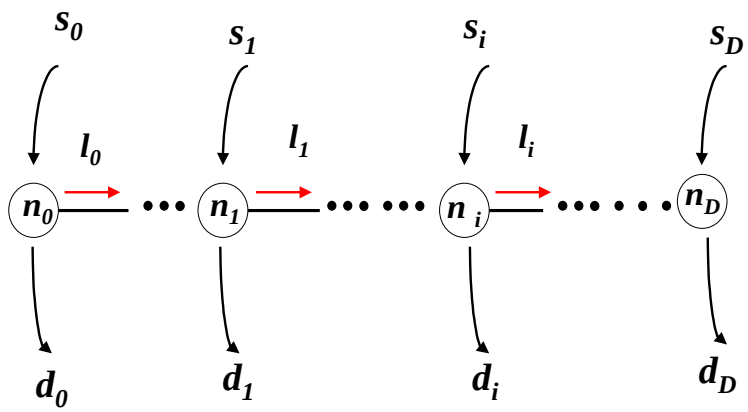


Figure 5: Model for analyzing the topological distribution of demand

Node n_i handles traffic in both directions, so the number of reservation flows traversing n_i is given by

$$N_i^{\text{RSVP}} = \sum_{j=0}^{i} s_j \sum_{k=i}^{D} d_k + \sum_{j=i}^{D} s_j \sum_{k=0}^{i} d_k - s_i d_i$$

for RSVP and

$$N_i^{\text{BGRP}} = \sum_{j=0}^{D} d_j$$

for BGRP.

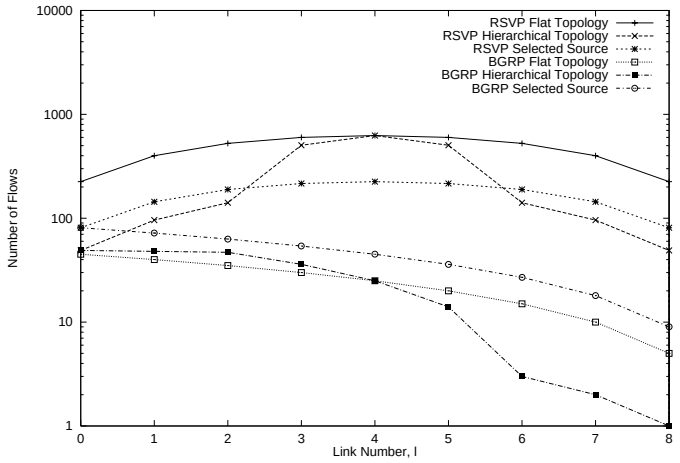
We now study both RSVP and BGRP in different networking scenarios, computing the number of flows and associated gains. We set $D = 9$, which is the maximum AS path length in the Internet today [21]. We simulated models with a total of 100 source and sink border routers. We assume that every source border router desires to set up a reservation to every sink border router.

Case 1: Flat Topology: The number of reservation sources and sinks are identical at each n_i; i.e., $s_i = d_i = 5$, for $0 \leq i \leq 9$.

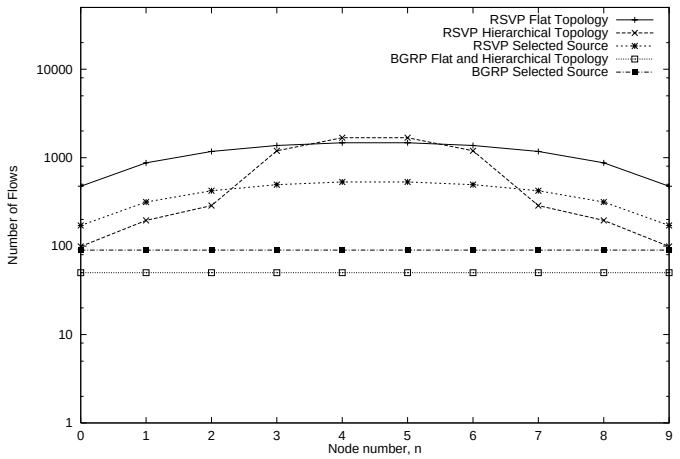
Case 2: Hierarchical Topology: We set $s_i = d_i = 11$, for $3 \leq i \leq 6$, and $s_i = d_i = 1$, for all other i. This models a hierarchical network where stub domains only contribute a small portion of the overall reservation flows, while most of the reservations are present at a few core transit domains.

Case 3: Selected Source: We set all s_i to 1, and all d_i to 9. This reflects a network where the number of data sources is small, but the number of sinks (data receivers) is large. This is a typical scenario for web applications.

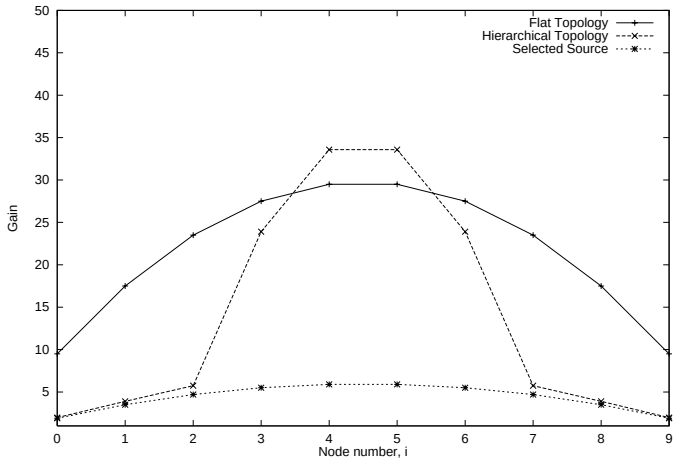
The results are shown in Fig. 6. Not surprisingly, BGRP maintains fewer reservations than RSVP. Fig. 6(c) shows that the largest gain occurs in the center of the network. Also, Fig. 6(b) shows that for



(a) Number of reservations per link



(b) Number of reservations per node



(c) The gain, $N_i^{\mathrm{RSVP}}/N_i^{\mathrm{BGRP}}$

Figure 6: Worst case scaling comparison between RSVP and BGRP.

RSVP, the number of reservations at a node depends on the node's topological location, whereas for BGRP, every node has the same number of reservations.

5.3 Reservation Dynamics

We now consider the effect of reservation dynamics in our performance analysis. For both RSVP and BGRP, we shall determine the control state overhead and the control message overhead as functions of the arrival rate of individual flows, the mean lifetime of a flow, and the protocol refresh interval. We assume that reservations are explicitly torn down when no longer needed, and that the blocking rate for reservations is negligible for our purposes.

Recall that the "virtual hop" between two "adjacent" border routers is called a BR-hop. We define the sequence of border routers visited by an end-to-end traffic flow as an edge-to-edge BR-path. For RSVP, each edge-to-edge BR-path establishes its own reservation. Let F be the total number of edge-to-edge BR-paths crossing a given border router R_i. For BGRP, these paths are aggregated into trees. Let T be the total number of sink trees formed by the paths through R_i. To keep this analysis simple, we assume that each sink tree at R_i is aggregating the same number $f = F/T$ of edge-to-edge BR-paths.

Now let us model the end-to-end flows desiring reservations. Any number of end-to-end reserved flows may be multiplexed onto a given edge-to-edge BR-path. We assume that the end-to-end reserved flows for one path arrive in a Poisson stream of rate λ, and that the flow reservation lifetimes are exponentially distributed with mean $1/\mu$. Then the number of reserved end-to-end flows on one edge-to-edge BR-path is Poisson distributed with mean $\rho = \lambda/\mu$, and the number of reserved end-to-end flows on any given sink tree is Poisson distributed, with mean $\rho \cdot f$.

Let us determine the state counts of the protocols. In case of RSVP, the probability that the edge-to-edge BR-path has a reservation, i.e., that at least one end-to-end flow is reserved, is $1 - e^{-\rho}$. Therefore, the number of reserved edge-to-edge BR-paths at R_i is binomially distributed with mean $(1 - e^{-\rho}) \cdot F$. This is the average amount of RSVP control state for R_i. For large ρ, this approaches F, while for small ρ, this approaches 0. Now let us determine the BGRP state count at R_i. The probability that a given sink tree has a reservation at R_i, i.e., that at least one end-to-end flow on at least one edge-to-edge BR-path on that tree has requested a reservation, is $1 - e^{-\rho \cdot f}$. The number of reserved trees on R_i is thus binomially distributed with mean $(1 - e^{-\rho \cdot f}) \cdot T = (1 - e^{-\rho \cdot F/T}) \cdot T$. This is the average amount of BGRP control state for R_i. For large ρ, this approaches T, while for small ρ, this approaches 0. We conclude that the state advantage of BGRP with respect to RSVP is more pronounced when ρ is large, where this gain approaches F/T. Figure 7 shows the mean number of simultaneous reservations for BGRP and RSVP as ρ ranges from 0.001 to 10; here we assume $F = 100,000$ and $T = 1,000$. For instance, when $\rho = 1$, the gain is 63. When $\rho = 10$, the gain has essentially reached its maxiumum value of 100.

Next we will compare the message rates for the two protocols. We will tally the control messages associated with reservations on one given unidirectional BR-hop. (Note that some of these "associated" messages actually travel on the reverse BR-hop.) When a new end-to-end flow for the given BR-hop is born, this counts as two messages in either protocol: PATH + RESV for RSVP, or PROBE + GRAFT for BGRP. Removing the reservation takes a single TEAR message, in either protocol. Since refresh is bidirectional, in both protocols, we count the refreshing of one reservation on a given BR-hop as two units of control message processing. (Refreshes for multiple reservations on the same BR-hop are assumed to be *processed* separately, even though they may be *transmitted* in a bundle.) Let η be the refresh rate. For RSVP, the average message rate for each edge-to-edge BR-path on the given BR-hop is $3\lambda + 2\eta \cdot (1 - e^{-\rho})$. Hence, the average RSVP message rate for the BR-hop is $[3\lambda + 2\eta \cdot (1 - e^{-\rho})] \cdot F$. For BGRP, the average message rate for one sink tree on the given BR-hop is $3\lambda \cdot f + 2\eta \cdot (1 - e^{-\rho \cdot f})$. Hence, the average BGRP message rate for the BR-hop is $[3\lambda \cdot f + 2\eta \cdot (1 - e^{-\rho \cdot f})] \cdot T$, which equals $3\lambda \cdot F + 2\eta \cdot T \cdot (1 - e^{-\rho \cdot F/T})$. If λ is much larger than η, then BGRP's PROBE, GRAFT and TEAR activities dominate its REFRESHes, and

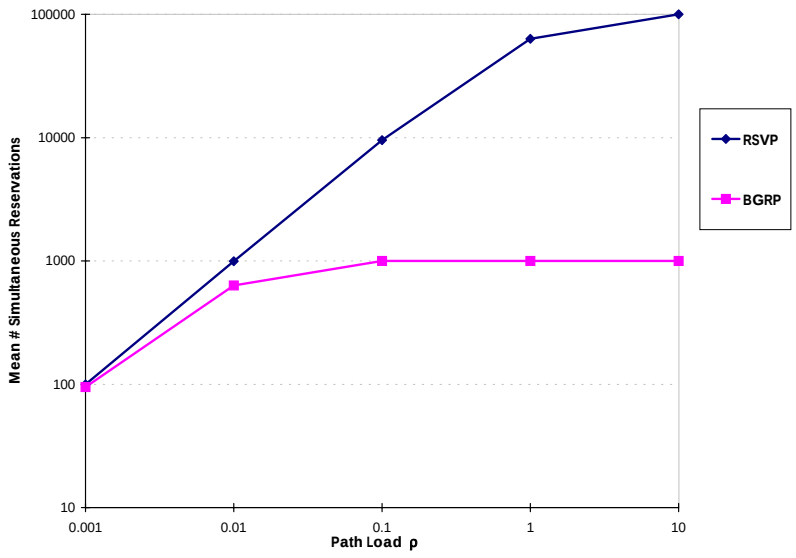


Figure 7: BGRP and RSVP reservation counts as functions of load ρ

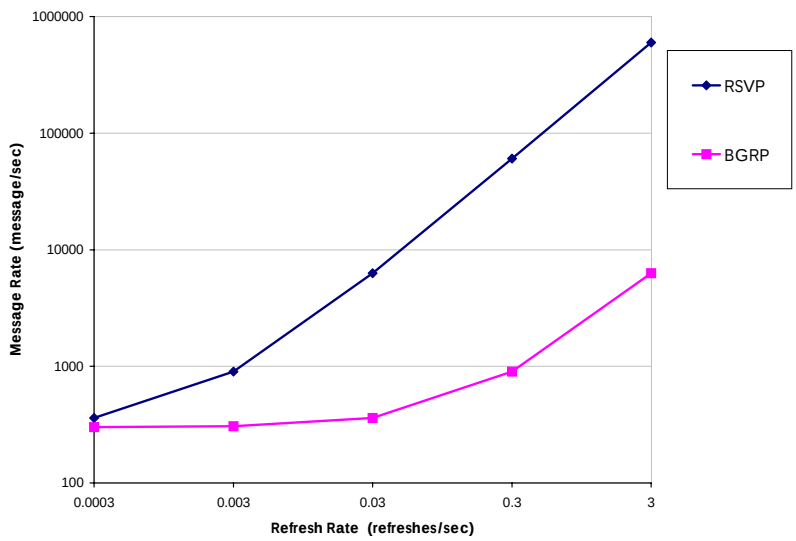


Figure 8: BGRP and RSVP message rates as functions of refresh rate η

RSVP's initial PATH and RESV messages dominate their refreshed versions. In this case, RSVP and BGRP have the same message rates. (Fortunately, the over-reservation techniques that we proposed in Sec. 4.1 can dramatically reduce BGRP's message processing overhead at this end of the spectrum; we will analyze these improvements shortly in Sec. 5.4 below.) On the other hand, if η is much larger than λ, so that REFRESH activity dominates, and if ρ is large, then BGRP does better than RSVP by a factor of F/T. Figure 8 shows the average message rates per second, for BGRP and RSVP, as η ranges from 0.0003 to 3.0 refreshes per second; here we assume $\lambda = 0.001$ flows per second, $F = 100,000$, $T = 1,000$, and $\rho = 10$. For instance, when $\eta = 0.03$ (i.e., when the refresh interval is about 30 seconds), then the gain (i.e., the ratio of RSVP messages to BGRP messages) is 18. When $\eta = 3.0$, the gain is 95.8.

5.4 Over-reservation, Quantization and Hysteresis

We showed in Section 5.3 that BGRP has fewer messages to process than RSVP, provided that most messages are REFRESHes rather than PROBEs, GRAFTs, and TEARs. In this section we show how hysteresis can be used to curb the PROBE, GRAFT and TEAR activity. The savings in message processing come at the cost of some wasted bandwidth, because the aggregate reservation sometimes exceeds the sum of the component flow requests.

We shall model a single aggregate reservation on a single BR-hop. We assume that the blocking rate for reservations is extremely low, negligible for the purpose of measuring mean message rates. Any number of end-to-end flow reservations can be multiplexed into this aggregate. Assume that these end-to-end reserved flows arrive in a Poisson stream of rate λ, and that the flow reservation lifetimes are exponentially distributed with mean $1/\mu$. Then the number of reserved end-to-end flows in our aggregate reservation is Poisson distributed with mean $\rho = \lambda/\mu$.

Assume that each end-to-end flow requires one unit of bandwidth. We constrain the aggregate reservation always to be a multiple of some quantum size $Q \geq 2$. Whenever the aggregate reservation is kQ units, and this is just barely enough to satisfy the current individual flows, and a new end-to-end flow reservation is requested, then the aggregate reservation jumps to $(k+1)Q$. This new quantum will only be relinquished when the sum of the individual flow requests drops to $(k-1)Q+1$. The amount of bandwidth wasted due to over-reservation by this technique is less than $2Q$ units.

We can model this system as a two-dimensional Markov process with state (x, y), where x is the number of currently reserved end-to-end flows, and y is the current aggregate reservation. Figure 9 shows the state transition diagram for $Q = 3$. The only valid states are the following: state (0,0), which we will ignore because it is transient; states (x, Q), where $0 \leq x \leq Q$; and states $(x, k \cdot Q)$, for $k \geq 2$, where $(k-2) \cdot Q + 2 \leq x \leq k \cdot Q$. The state transition diagram has a structure that makes it straightforward to solve for the steady-state probabilities. Note that, for most values of x, there are two possible values of y, i.e., two states. However, certain special values of x have only one possible y value, i.e., one state. These special values of x are: $x = 0$, for which y must equal Q, and $x = Q+1, 2Q+1, 3Q+1, ...$, for which the respective values of y must be $2Q, 3Q, 4Q, ...$ Therefore, at these special points, the joint probability distribution $\pi(x, y)$ matches the marginal distribution of x by itself, which we already said is Poisson. It is straightforward to determine the probabilities of all the other states in terms of these special states, thereby completing the analysis. For all $k \geq 1$:

$$\pi(0, Q) = e^{-\rho}$$

$$\pi(((k-1)Q+1), kQ) = \frac{e^{-\rho} \cdot \rho^{((k-1)\cdot Q+1)}}{((k-1)\cdot Q+1)!}$$

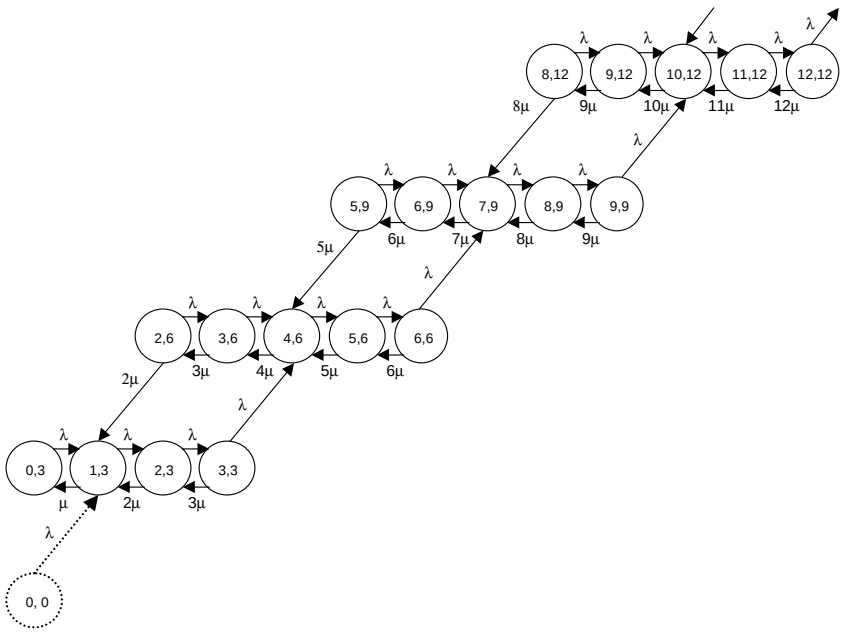


Figure 9: State transition diagram for $Q = 3$

For $(k-1)Q+2 \leq x \leq kQ$:

$$\pi(x, k \cdot Q) = \frac{e^{-\rho} \cdot \rho^x \cdot \sum_{i=0}^{k \cdot Q - x} [\rho^i \cdot (k \cdot Q - i)!]}{x! \cdot \sum_{i=0}^{Q-1} [\rho^i (k \cdot Q - i)!]}$$

$$\pi(x, (k+1)Q) = \frac{e^{-\rho} \cdot \rho^x \cdot \sum_{i=k \cdot Q+1-x}^{Q-1} [\rho^i \cdot (k \cdot Q - i)!]}{x! \cdot \sum_{i=0}^{Q-1} [\rho^i \cdot (k \cdot Q - i)!]}$$

The rate of BGRP control messages (PROBEs + GRAFTs + TEARs) is:

$$3\lambda \cdot e^{-\rho} \cdot \sum_{k=1}^{\infty} \left(\frac{\rho^{(k \cdot Q)}}{\sum_{i=0}^{Q-1} [\rho^i \cdot (k \cdot Q - i)!]} \right)$$

This compares very favorably to the message rate 3λ without hysteresis. Figure 10 shows the message rate reduction factor for various values of Q and ρ. For example, if reservations are quantized in blocks of 10, and $\rho = 100$, then quantization and hysteresis reduce the BGRP message rate by a factor of about 100. (The curves in Fig. 10, while roughly decreasing, are not strictly monotonic. The ripples in the plots are due to the many "corners" in the state transition diagram of Fig. 9. For a given quantum size Q, as ρ increases, the bulk of the probability mass zigzags upward and to the right through the state space. Since vertical state transitions produce protocol messages while horizontal transitions do not, these zigs and zags around the corners can produce ripples in the messaging efficiency plot.)

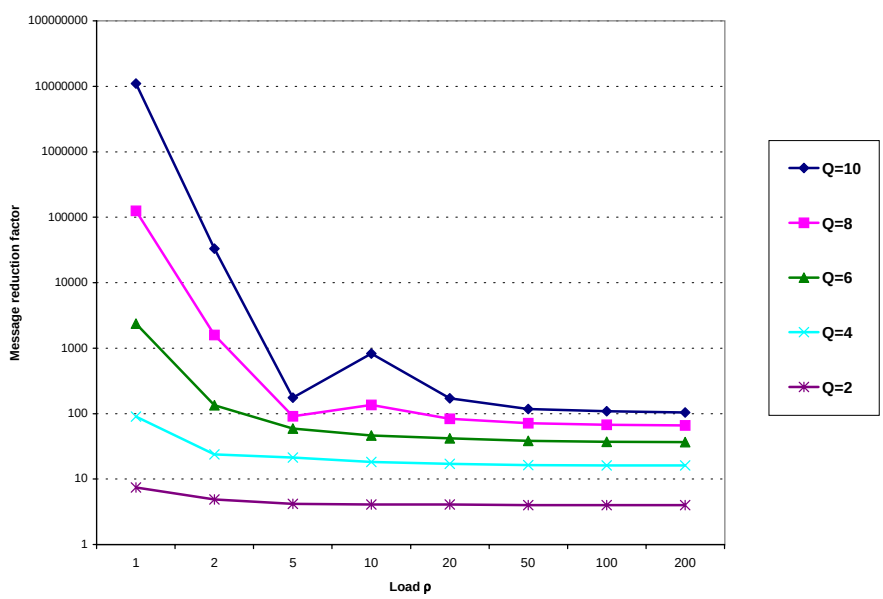


Figure 10: Message reduction factor as function of Q and ρ

The analysis above dealt with a single aggregate reservation on a single BR-hop. It is a good model for

a system where the *leaves* of sink trees are the only BR-hops that can initiate an over-reservation. However, if BR-hops *anywhere* on the tree can deliberately over-reserve, and if the quiet grafting described in Sec. 4.2 is done, then additional savings in message rate are possible.

Note that, in order for over-reservation to be helpful, the over-reserving router must recognize the future traffic on whose behalf the over-registration was made, without having to send a new PROBE message. For the tree leaves that over-reserve, simple caching of the destination network IDs associated with each tree would work. For more complex situations, see the labeling discussion in Section 4.2.

6 Related Work

Recently, several authors have addressed scalable resource reservation, using either a server-based or a router-based approach.

In server-based approaches, each domain has a bandwidth broker (or agent) which is responsible for selecting and setting up the aggregated reservation sessions. This approach has the advantage of removing the message processing and storage burden from routers. However, synchronizing reservation information among the bandwidth brokers and the routers may be complex. No aggregation takes place, so that each broker still has to deal with the requests of individual flows. Also, care has to be taken so that the broker does not become a single point of failure for the domain. Variations of the server-based approach have been described by Blake *et al.* [6], Schelen and Pink [22], Berson *et al.* [23], and Reichmeyer *et al.* [24]. The latter proposal suggests a two-tier system where, within each domain, hosts use intra-domain reservation protocols such as RSVP to set up reserved flows. Inter-domain reservation protocols set up coarsely-measured reserved flows between domains. However, the proposal leaves the actual mechanism undefined.

Awduche *et al.* [25], Guerin *et al.* [26] and Baker *et al.* [27] have proposed a router-based approach by modifying RSVP to support scalable reservation. (Awduche's LSP tunnels [25] are designed to support *intra*-domain traffic engineering, but may also be used to set up trunks crossing multiple domains.) These proposals aggregate per-application reservation requests into reservation "trunks" between pairs of domains, by modifying sender template and session objects in RSVP to carry address and mask ("CIDR blocks") or autonomous system (AS) numbers instead of 5-tuples (sender IP address, sender port, receiver IP address, receiver port, protocol). However, this implies that routers in the backbone may have to maintain reservation state proportional to the square of the number of CIDR blocks or autonomous systems. Since the number of AS is currently about 6,000, the number of AS pairs exceeds 36,000,000! As we shall argue in Sections 2 and 5.1, 36 million is too many. A more aggregated reservation scheme is needed.

In addition, Feher *et al.* [28] has most recently proposed a state-less reservation mechanism, Boomerang, where end users send reservation requests and refresh messages to set up and consequently maintain reservations. No per-flow state is stored at routers. However, this mechanism may have scaling implication due to overall control message processing cost.

7 Conclusions and Future Work

This paper presented a distributed architecture and protocol for inter-domain resource reservation, in which reservations are aggregated along sink trees. We showed that our BGRP scheme scales well in terms of protocol processing, protocol bandwidth, protocol state, data forwarding mechanics, and inter-domain administration:

Protocol Processing. Control message processing is the most important scalability issue. The cost of processing reservation messages depends on many things, among them: the volume of requests for new

reservations, the volume of existing reservations requiring periodic refreshing, and the refresh frequency. BGRP economizes on all of these components. First, when we allow routers to over-reserve bandwidth with BGRP, then small reservations can join and leave the reservation sink tree without disturbing the entire tree. Instead of setting up reservations across domains as application flows arrive, we can rely on bulk reservations made in advance. This reduces the volume of reservation set-up messages. Second, by aggregating reservations, BGRP reduces their number, and this proportionally reduces the refresh processing burden. Third, BGRP needs less frequent refreshes than does RSVP, for the following reason. RSVP control messages are unreliable and thus must be repeated at about three times the state-timeout interval, while BGRP refresh messages are transferred reliably hop-by-hop.

Protocol Bandwidth. The mechanisms described above reduce the number of control messages. Not only does this reduce the burden on the routers to process these messages, it also reduces the burden on the links to transmit the messages.

Protocol State. By aggregating reservations, BGRP reduces their number, and thereby reduces the memory needed to store the control information. The number of BGRP reservations in a router is bounded by either the number of domains or the number of announced routing prefixes, depending on the version of the protocol used. Currently, the Internet encompasses approximately 6,000 domains and 60,000 network addresses. Assuming that, as in RSVP, each reservation occupies 350 bytes of control memory [1], then BGRP state consumes at most 2.1 MB for AS-level reservation, or 21 MB for network-level reservation. This is far less than the storage space for routing protocols; e.g., BGP routing tables can occupy hundreds of megabytes of memory in a backbone router.

Data Forwarding. BGRP is designed to work within the DiffServ QoS framework. There is no flow-based packet classification, queueing, or scheduling in backbone routers. Hence, the data forwarding operations are quite scalable.

Administration. BGRP operates on the boundaries between ISPs, leaving each ISP free to manage its own domain independently. BGRP can be deployed easily because it does not require changes to local resource management.

This paper presented the basic architecture, protocol design, and performance analysis. In the future, we will continue to explore various BGRP enhancements, such as reservation route pin-down, and to examine various options for rooting and labeling the reservation sink trees. We will also consider several options for PROBE message processing: whether resource availability really needs to be checked during this first protocol phase, and if so, whether the resource should also be reserved at that time. More dramatic changes, such as the use of source trees (instead of sink trees) and support for multicast, may also be considered. At the same time, we are implementing BGRP on backbone router platforms and will test it in realistic networking environments. We expect to propose BGRP to the IETF shortly.

8 Acknowledgements

Sean McCreary helped us analyze the AS traces. Craig Labovitz advised us on various aspects of network operations that are relevant to BGRP. Tony Przygienda and Rohit Dube commented on the interaction between BGP and routing aggregation. We also thank Fred Baker, Kimberly Claffy, Der-Hwa Gan, George Swallow, and Andreas Terzis for many discussions leading to this work.

References

[1] P. Pan and H. Schulzrinne, "YESSIR: a simple reservation mechanism for the Internet," *ACM Computer Communication Review*, vol. 29, pp. 89–101, Apr. 1999.

[2] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: a new resource ReSerVation protocol," *IEEE Network*, vol. 7, pp. 8–18, Sept. 1993.

[3] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation protocol (RSVP) – version 1 functional specification," Request for Comments (Proposed Standard) 2205, Internet Engineering Task Force, Sept. 1997.

[4] T. cker Chiueh, A. Neogi, and P. Stirpe, "Performance analysis of an RSVP-capable router," in *Proc. of 4th Real-Time Technology and Applications Symposium*, (Denver, Colorado), June 1998.

[5] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview," Request for Comments (Informational) 1633, Internet Engineering Task Force, June 1994.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service," Request for Comments (Informational) 2475, Internet Engineering Task Force, Dec. 1998.

[7] J. Agogbua, D. Awduche, J. Malcolm, J. McManus, and M. O'Dell, "Requirements for traffic engineering over MPLS," Internet Draft, Internet Engineering Task Force, June 1999. Work in progress.

[8] Internet Software Consortium, "Internet domain survey." http://www.isc.org/ds/.

[9] Telstra Network, "Telstra internet network performance reports." http://www.telstra.net/ops/.

[10] T. Bates, "The cidr report." http://www.employees.org/~tbates/cidr-report.html.

[11] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet routing instability," in *SIGCOMM Symposium on Communications Architectures and Protocols*, (Cannes, France), Sept. 1997.

[12] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP-4)," Request for Comments (Draft Standard) 1771, Internet Engineering Task Force, Mar. 1995.

[13] D. J. Mitzel and S. Shenker, "Asymptotic resource consumption in multicast reservation styles," in *SIGCOMM Symposium on Communications Architectures and Protocols*, (London, UK), pp. 226–233, Sept. 1994.

[14] L. Berger, D. Gan, G. Swallow, and P. Pan, "RSVP refresh reduction extensions," Internet Draft, Internet Engineering Task Force, July 1999. Work in progress.

[15] C. Labovitz, G. Malan, and F. Jahanian, "Origins of internet routing instability," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (New York), Mar. 1999.

[16] P. Pan and H. Schulzrinne, "Staged refresh timers for RSVP," in *Proceedings of Global Internet*, (Phoenix, Arizona), Nov. 1997. also IBM Research Technical Report TC20966.

[17] C. Villamizar, R. Chandra, and R. Govindan, "BGP route flap damping," Request for Comments (Proposed Standard) 2439, Internet Engineering Task Force, Nov. 1998.

[18] NLANR, "Nlanr network traffic packet header traces." http://moat.nlanr.net/Traces/.

[19] T. Li and Y. Rekhter, "A provider architecture for differentiated services and traffic engineering (PASTE)," Request for Comments (Informational) 2430, Internet Engineering Task Force, Oct. 1998.

[20] K. Nichols, V. Jacobson, and L. Zhang, "A two-bit differentiated services architecture for the internet," Internet Draft, Internet Engineering Task Force, May 1999. Work in progress.

[21] NLANR, "Nlanr as path lengths." http://moat.nlanr.net/ASPL/.

[22] O. Schelen and S. Pink, "Aggregating resource reservation over multiple routing domains," in *Proc. of Fifth IFIP International Workshop on Quality of Service (IwQOS)*, (Cambridge, England), June 1998.

[23] S. Berson and S. Vincent, "Aggregation of internet integrated services state," in *submitted to IWQOS*, 1998.

[24] F. Reichmeyer, L. Ong, A. Terzis, L. Zhang, and R. Yavatkar, "A two-tier resource management model for differentiated services networks," Internet Draft, Internet Engineering Task Force, Nov. 1998. Work in progress.

[25] D. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, and V. Srinivasan, "Extensions to RSVP for LSP tunnels," Internet Draft, Internet Engineering Task Force, Mar. 1999. Work in progress.

[26] R. Guerin, S. Herzog, and S. Blake, "Aggregating RSVP-based QoS requests," Internet Draft, Internet Engineering Task Force, Nov. 1997. Work in progress.

[27] F. Baker, "Aggregation of RSVP for IP4 and IP6 reservations," Internet Draft, Internet Engineering Task Force, June 1999. Work in progress.

[28] G. Feher, K. Nemeth, and M. Maliosz, "Boomerang - a simple protocol for resource reservation in ip networks," in *IEEE Workshop on QoS Support for Real-Time Internet Applications*, (Vancouver, Canada), June 1999.