MPSS: Multiprotocol Stateless Switching

András Zahemszky*†, Petri Jokela*, Mikko Särelä*, Sami Ruponen‡, James Kempf* and Pekka Nikander*
*Ericsson Research
Email: firstname.lastname@ericsson.com
†Helsinki Institute for Information Technology
‡VTT Technical Research Centre of Finland
Email: sami.ruponen@vtt.fi

***Abstract*—The Multiprotocol Label Switching (MPLS) architecture has become a true success story in the world of telecommunications. However, MPLS becomes cumbersome if multicast communication is needed, as aggregating of labels is not easy. Because of that, when providing Multicast VPNs, operators need to trade-off bandwidth usage with the amount of multicast state, sacrificing efficiency. Forwarding with Bloom filters in the packet headers offers the opportunity to have quasi-stateless network elements; the amount of forwarding plane state does not depend on the number of paths/trees the node participates in.**

In this paper, we propose Multiprotocol Stateless Switching (MPSS), the marriage of MPLS and Bloom filter based forwarding. The forwarding architecture inherits the flexibility of MPLS and gives operators the opportunity to offer Multicast VPN services while avoiding the difficult process of fine-tuning the trade-off between bandwidth usage and state.

I. Introduction

Saltzer [1] defines four types of objects that can be named as destinations of packets in networks: services and users, nodes, network attachment points, and paths. The Internet Protocol names network attachment points and routing protocols ensure that routers know the path towards these attachment points.

In pure IP, there does not exist any name for *paths*. MPLS [2] changed this by introducing a name, a fixed sized label, for them[1]. The routing protocol (with label distribution) ensures that the edge nodes know the label to use for a given destination and the intermediate nodes merely follow the path instructions in the header.

The original motivation for MPLS was to increase the forwarding speed, but this benefit has nowadays a lower importance as the IP packet forwarding is done in hardware. But in addition to speed, MPLS provides extra benefits, such as Traffic Engineering and support for various applications, such as VPNs, or Pseudo Wires.

However, assigning a label for each of the paths used in the network, will result in an explosion in the label space. Moreover, with multicast, the number of potential trees is huge when compared to the number of nodes.

In our prior work, we introduced zFilters [3], a novel way of forwarding packets in networks. The presented mechanism encodes a path/tree into a small Bloom Filter. The nodes in the network need only be aware of their neighbors in the forwarding plane, minimizing the size of the forwarding table on a single router.

[1]To be exact, each node on the path has a separate label for the path and instructions how to change the label for the next router.

In this paper, we propose a marriage between the MPLS and zFilter technologies, called Multiprotocol Stateless Switching (MPSS). MPSS inherits the flexibility of MPLS and enhances it with the stateless nature of the zFilter based forwarding.

The paper is structured as follows. In Section II, we give an overview of MPLS and zFilters. In Section III, we discuss how to set up paths and trees in MPSS. In Section IV, we show how MPSS can be used for providing Multicast VPN services for customers. We briefly describe our on-going implementation efforts in Section V, give an overview of the related work in Section VI, and finally, conclude the paper in Section VII.

II. Background

In this section, we give an overview of the technologies we are building on, first focusing on the MPLS framework, and summarizing the Layer 3 VPN architecture. After that, we describe the basic design of our Bloom filter-based forwarding architecture.

A. *MPLS framework*

MPLS separates routing and forwarding. Routing decisions are made at the edges of the network and a resulting label, describing the path in the network, is attached to the packet. The label is used by the intermediate routers to forward the packet. In the simplest case, the outgoing interface is determined using the incoming interface and the packet's label, and the label is swapped for the next hop. It is also possible to stack the labels, where nodes can perform push and pop operations.

Label Switched Paths (LSPs) can be set up using either LDP [4] or RSVP-TE [5] as the signalling protocol. In LDP (Label Distribution Protocol), the routers assign a label to each IP destination found in their forwarding table and communicate it with their neighbors. Consequently, multipoint-to-point tunnels are built and these tunnels always follow the shortest paths defined by the underlying routing architecture.

Contrary to the LDP, RSVP-TE (Resource Reservation Protocol with Traffic Engineering) is used when Traffic Engineering (TE) is required for optimizing the internal network usage. TE related tasks include the capability of signalling explicit paths, enforcing bandwidth guarantees, and also, providing QoS, by being capable of differentiating flows.

In OSPF-TE and IS-IS-TE, the characteristics of the links, such as available bandwidth, are advertised. When an LSP is needed, the path is computed either by the ingress node, or by the Path Computation Element (PCE) [6], using the available advertised information and the requirements for the path. The algorithm is called as CSPF (Constrained Shortest Path First). Once calculated, the path is signalled hop-by-hop, first downstream with *Path* messages and then upstream with *Resv* messages. The on-path routers perform admission control, assign labels, reserve resources, and set the forwarding tables.

In some cases, the LSP may be broken and has to be recovered. For this, RSVP-TE defines mechanism for signalling backup paths and reserving resources if needed.

Finally, MPLS became the enabler of various applications, due to its flexibility and its multiprotocol nature. Different Virtual Private Network (VPN) solutions (L2VPN, L3VPN, VPLS) appeared based on MPLS. Furthermore, Pseudo Wires (PWs) are used to transport various L2 technologies. All these applications can be handled by a single, multi-service infrastructure.

B. Layer 3 Virtual Private Networks

The operator's network, providing L3VPN [7] connections to customers, consists of Provider Edge (PE) and Provider (P) routers (see Figure 1). PEs maintain a Virtual Routing and Forwarding Table (VRF) for each VPN they are serving. Each VRF contains the corresponding VPN routing information. BGP is used between PEs to exchange the VRF information. Using specific attributes, the VPNs can have overlapping address spaces and routes are distributed to the correct VRFs.

When forwarding unicast IP packets, the ingress PE attaches two MPLS labels to the packets. The outer label defines the path to the egress PE through the provider's network. The inner label, in turn, is a service label identifying any additional action in the egress PE, e.g. the correct VPN, or outgoing link. The value of the inner label is advertised by BGP.

For multicast, several options are being specified in IETF [8], both in the control and in the data plane. The control plane function of distributing the multicast group membership information through the provider's network is done either with PIM, or via new BGP extensions.

In data plane there are several options; we can use plain IP multicast, ingress replication (multiple unicast), point-to-multipoint or multipoint-to-multipoint LSPs. Still, all these approaches have to make trade-offs between bandwidth usage and amount of state. In contrast to unicast, some solutions imply VPN-aware P routers to avoid unacceptably high amount of state when optimizing the bandwidth. Clearly, if all customer multicast group (or each combination of PE nodes) has a dedicated tree in the core, the bandwith usage can be minimalized. Reducing state requirements trees will cause some bandwidth waste; some packets will be routed to PEs, that do not have receivers for the traffic.

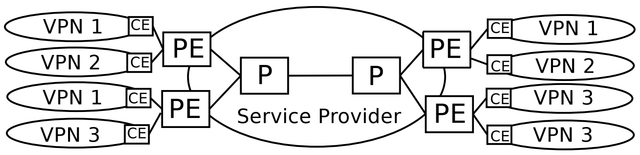
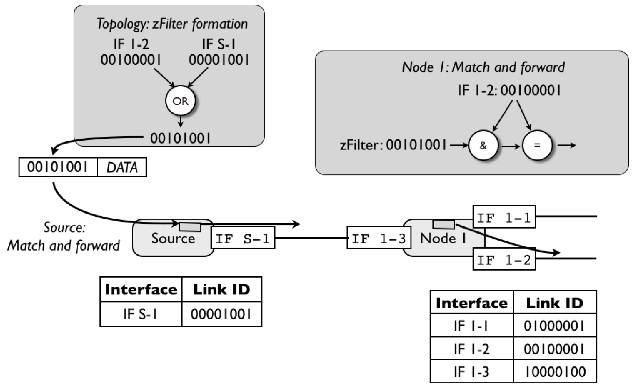


Fig. 1. L3VPN scenario

C. Source Routing with In-packet Bloom Filters

As shown in LIPSIN [3], source routes and trees can be efficiently encoded into the packet header using Bloom filters [9] and locally naming links instead of nodes. Once the path or tree is determined, the forwarding identifier, called zFilter (zF), is formed by compressing the set of link identifiers into a Bloom filter.

A link identifier is a fixed length, m-bit long string, with k bits set to one, where $k \ll m$, and m is relatively large. With $m = 256$ and $k = 5$, we get $\approx m!/(m-k)! \approx 10^{12}$ different link identifiers, making link identifiers statistically unique (assuming the k set bits are randomly distributed). Each node has a function $Z(L, I)$, which is used to compute the link identifier based on local information L and some information I from the packet header. We describe two variants of the Z-function at the end of this section based on LIPSIN [3] and Zformation [10].



Interface	Link ID
IF S-1	00001001

Interface	Link ID
IF 1-1	01000001
IF 1-2	00100001
IF 1-3	10000100

Fig. 2. Forwarding with zFilters

To construct a zFilter, a binary OR over all the link identifiers forming the tree T is computed (see Fig. 2). The source uses the resulting zFilter Z_T to send packets along the delivery tree T.

The forwarding node makes a forwarding decision for each of the outgoing link o by checking if the zFilter Z_T contains the Link Identifier L_o, i.e. if $(Z_T \wedge L_o) \equiv L_o$. If this is the case, the packet is sent out of that interface. If the zFilter contains multiple outgoing Link IDs, then the packet is forwarded to each of them, resulting in multicast.

Bloom filters have false positives. In zFilter forwarding, this results in some additional packets to be delivered in the network, typically over one link. The probability of this increases when more links are added to a zFilter. For quantitative analysis, we refer to [3]. Shortly, the analysis there revealed that around 35-40 links can be placed into a 256-bit zFilter to achieve at least 90% forwarding efficiency (the rate of useful traffic).

A simple $Z(L, I)$ merely has a static pre-computed link identifier L for each link. As we show in [3], the number of false positives can be reduced using for each outgoing link a set of possible link identifiers and allow the packet to carry a value that decides which one of them to use. In [10], we show that storing and using a key as the local input value, we can achieve constant sized forwarding table and increase security. For example, the link identifier calculation can be based on the flow information (IP 5-tuple), the incoming interface, and the actual key. This means that the zFilter is only valid on its intended path, only for a specific time, and only with the given 5-tuple.

III. MPSS ARCHITECTURE

In this section, we describe both IP and zFilter-based signalling mechanisms for setting up stateless, multicast-friendly, MPLS-like forwarding structures. Three tasks should be handled by the following signalling processes: a) computing a path/tree on the graph of the network, b) computing a zF representation of it, and optionally, c) allocating resources.

Path/tree computation can be handled by existing MPLS technologies. The zF creation can either be co-located with the tree computation or alternatively, we can use a reverse path zFilter collection (zF-c) field in the signalling messages. In an initially empty zF-c field, routers add the incoming interface's link IDs by ORing it with the existing zF-c. Finally, resource allocation can be done with an RSVP process.

A. Merging zFilters and MPLS

MPLS packet forwarding requires state in the network. The unicast path setup is trivial, but with multicast the tree setup and modification gets very complex. By replacing MPLS labels with zFilters, the data plane is stateless and natively supports multicast. Thus, we can avoid a lot of complexity, especially with multicast.

In MPLS, connection setup with RSVP is done by a two way message exchange on the explicit path of the LSP. This same signalling exchange is used for both reserving resources and setting up forwarding tables. When using zFilters, the forwarding tables can be left untouched. The resource reservation requires some minor modifications, mainly in how the packet is handed from the forwarding layer to the control plane processing.

B. Resource allocation in MPSS

Resource allocation is one fundamental building block for Traffic Engineering. In this section, we propose extensions to RSVP-TE to be able to handle zFilters. We describe two alternatives; in one alternative, the signalling messages are routed by IP; in the other, by zFilters.

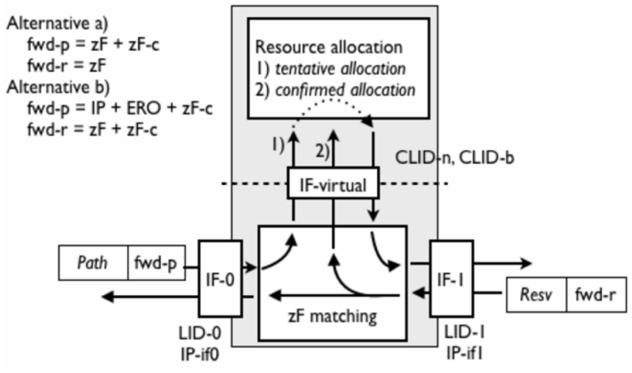


Fig. 3. Resource Allocation at P router

In Figure 3, the RSVP operation on a P node is depicted. When the RSVP Path message enters the P node, it is handed to the control plane, where the requested resource is tentatively allocated, and the packet is forwarded further. At the egress PE, the RSVP Resv packet is sent for confirming the tentative allocations. If the requested resource is not available on some router, the Path packet is not forwarded, but a failure notification (PathErr) is sent back.

While with IP-based signalling (Figure 3, Alt. b) the packet is sent hop-by-hop according to the path described in the ERO object, with zF-based signalling (Alt. a) we use two types of node-internal, well-known control LIDs: Blocking (CLID-b) and non-blocking (CLID-n). When a CLID-b is included, the packet is not forwarded out of the node until approved by the control layer, but with CLID-n, the packet is multicasted simultaneously to the control layer and to all outgoing interfaces matching the zF. For all Path- ackets, the CLID-b is ORed with the zF, and for reverse direction packets (Resv, PathErr), the CLID-n is included.

C. Setting up MPSS paths

In the following proposals, the control of the tree setup is always at the ingress PE, making it more flexible to add or remove receiving PEs from the point-to-multipoint delivery tree than with a solution relying on branch points inside the network. This requires that the source should be able to store the unicast zFilters of each branch to egress PEs on the tree. The multicast zF is computed by the simple method of ORing together the unicast zFs.

Due to broken links, or other events, it may be possible that the data traffic needs to be re-routed. To handle these cases, we can use the methods defined in [11].

zFilter Distribution Protocol. Like in LDP's ordered control, PEs advertise themselves to their neighbors. Each router, receiving a PE advertisement, forwards it further to its neighbors if the packet arrived via the shortest path according to its IP routing information. Each advertisement contains a zF-c field and other PEs will store the collected unicast zF from the received advertisement. Using these unicast zFs, the PEs can compute also any point-to-multipoint shortest path trees.

The ingress PE initializes resource allocation (see Figure 3), by sending the Path packet towards the egress PE, with CLID-b included in the zF. Whenever resource allocation is not needed, the computed zF can be used immediately.

Path Request with RSVP is initiated by the ingress PE with a Path packet, containing the explicit path (computed locally or remotely) in the ERO object and a zF-c field (Figure 3, alt. b). Once the Path arrives to the egress PE, it will reply with the Resv packet, using the collected zF. This packet contains also the zF-c field collecting the data forwarding zF. As for point-to-multipoint trees, each branch has to be requested separately, resulting in unicast zFs to the egress PEs, which should be ORed together in the last step.

However, if there is no need for resource allocation, the requested bandwidth is set to zero, and there is no need for the Resv packet to be handled at the control layer.

The Existing Topology Information, collected e.g. with OSPF-TE, can be used by the ingress PEs to calculate zFs to other PEs, without a need for an explicit setup packet. The Link ID information needed for zF creation can be either be distributed using a modified OSPF-TE, or alternatively, calculated using a well-known algorithm.

If there is no need for resource reservation, the calculated zFs can be used immediately for data traffic. Otherwise, the ingress PE should initiate the RSVP process.

IV. MPSS based Multicast VPNs

In this section, we show a use case for MPSS. We discuss how Multicast VPN services can be offered by installing MPSS into the service provider's network. The reference architecture is shown in Fig. 1. The traffic in the customer sites is IP traffic; zFilters are used for forwarding in the provider's network.

Similar to current designs, we employ a hierarchy of labels (see Section II-B). In our system, the outer label is a zFilter, while for the inner labels we keep the MPLS label.

A. Control plane

The control plane is responsible for two tasks. The first task is to build paths and trees inside the provider's network. Sec III discusses a few different options.

The other task is to advertise the customers' routing information through the provider's network. As we use MPLS labels for demultiplexing in the egress PEs, the control plane of current VPN deployments can be readily used. For advertising unicast routes, BGP does not require any further extensions. As a result, PEs will distribute the routes in the customer networks, and the VRFs will be correctly set.

In the multicast case, from the options the current specifications offer, a membership distribution protocol should be chosen which allows explicit tracking of receivers. The source PEs will be always aware of the receiver PEs for a given multicast group, and therefore they can always use an appropriate zFilter for forwarding inside the network.

B. Data plane

When a PE receives a unicast IP packet destined to a remote site, it checks its corresponding VRF, and determines the egress PE and the inner label. It attaches the inner label to the packet, and a zFilter to reach the specific PE. The P routers then forward the packet based on the zFilter. When the egress PE is reached, it removes the zFilter and checks the MPLS label. Based on the decision, it forwards the original IP packet to its correct destination network.

Multicast works in a similar way. When receiving an IP multicast packet destined to remote networks, the PE labels it with an inner label communicated by the control plane and a zFilter to reach the PEs with receivers, or alternatively, the PE that serves the rendezvous point for the particular group.

C. Pushing back secured forwarding state to the CE

Typically, a single PE router serves multiple customer networks and therefore the routing decisions and encapsulation would impose huge load that might not be acceptable in some settings (eg. consider a PE with serving 100 separate VPNs). To reduce the load, the PE may pass back the computed zFilter to the CE. Once the CE receives the zFilter, it is be responsible for mapping the IP 5-tuple to the zFilter, and labelling the packet.

Now, the packet will already contain the zFilter and the inner label when it travels through CE-PE link. Consequently, the packet can be forwarded in the PE through the fast path.

The zFilter needs to be computed based on the IP 5-tuple and also, based on the VPN label. This ensures that a malicious CE cannot send packets to destinations, other than the one it got the permission from. As the zFilter is also bound to the VPN label, the CE cannot send packets to other VPNs. Additionally, the zFilter calculated for the customer flow can also depend on the QoS bits in the IP header. In an attempt of an attack (ie. modifications in the packet header), the packet is quickly discarded without reaching the destinations.

Every time when the routing information for a specific unicast route or for a multicast group changes (receivers join or leave), the information pushed back to the CE should be renewed. Upon change, the PE may install a flow filter that matches the old zFilter. A match here causes the packet to be processed on the slow path.

D. Discussion

The main benefit of using zFilters in the data plane that P routers can be quasi-stateless. Their number of state entries does not depend on the number of PE routers, neither on the number of (Multicast VPNs), nor the number of trees they participate in. As shown in [3], the quasi-statelessness has no penalty in unicast and sparse multicast traffic with 256-bit long zFilters. Also, it comes with acceptable bandwidth inefficiency until 20 receivers in large operator networks. These results, together with the measurements carried out in [12], show that MPSS can be an attractive alternative to MPLS for deploying Multicast VPNs.

With the ability to push back the result of the routing decisions to CEs, costly upgrades of PE routers can be delayed, as the processing load on them will be reduced. With the ideas from [10], we can ensure that the zFilter and the VPN label cannot be modified without being noticed.

V. Early Implementation Efforts

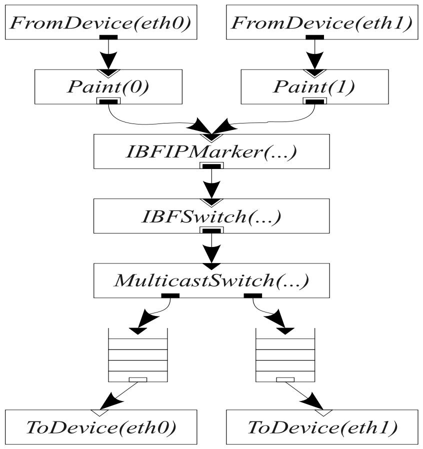


Fig. 4. Elements implemented in Click

Currently we are implementing the data plane on Click [13]. Figure 4 shows the router configuration that models the behavior of a simple P router. *FromDevice* and *ToDevice* elements are the input and output interfaces of the router. Incoming packets are painted (*Paint*) according to input interface. *IBFIPMarker* annotates the packet with the position of flow information, in this case the IPv4 source and destination addresses. *IBFSwitch* uses the flow information and the incoming port annotations and the actual key (K) and also reads the actual zFilter from the packet. Based on these, the outgoing LID is calculated and matching outputs are annotated to the packet. *MulticastSwitch* reads the output annotation and replicates the packet to specific output queues.

VI. Related work

One category of related work contains the works that aim to reduce label consumption in MPLS networks, and the other contains some relevant alternative forwarding architectures.

Label space reduction. RFC 5439 [14] presents a thorough scaling analysis in MPLS networks, concerning the amount of unicast LSPs and forwarding plane state. The analysis shows that in large networks traffic engineered multipoint-to-point (mp2p) LSPs are desirable, in contrast to hierarchical LSPs and traditional label swapping. In [15], the goal is to create a minimal number of mp2p LSPs from a given set of unicast ones. Though this is NP-complete, online and offline merging algorithms were given, and the label space usage was reduced to 30% in the offline case (with worse results for the online scenario). An offline algorithm is proposed in [16], based on mp2p trees, and strict upper bounds for the minimum number of labels required are proved. If more than one label is allowed in the packet, algorithms can be designed to minimize the number of labels for a given label stack size, or for a given label space to minimize the stack size [17].

Edge Router Multicasting [18] is a technique to build point-to-multipoint (p2mp) trees from unicast LSPs in a way that branching only occurs at the edge nodes. While the state requirements in the core are the same as in the unicast cases, the bandwidth usage cannot be optimal in this method. Solano et. al propose the usage of asymmetric tunnels, and an offline [19] and an online algorithm [20] based on them for reducing the label space usage of p2mp LSPs. Finally, they reduce packet overhead by combining the concept of tunnels with label merging in [21], requiring a single label per packet. Further example of label aggregation for multicast is [22].

The work in [23] summarizes the mechanisms of traffic aggregation in MPLS-based VPNs, and presents a simple analytical model for the bandwidth and state trade-off. As a difference, in MPSS each multicast group in each VPN can have its own tree in the network.

Compared to the related work, we use longer packet headers to be able to drastically reduce the label space usage. Recently, it has been argued that longer packet headers are not necessarily infeasible [24]. Also, the label is unchanged when the packet is forwarded in the core.

Alternative routing and forwarding architectures. The authors in [25] argued to use IP for forwarding in the VPN scenarios. However, in multicast, the intrinsic problems of state and bandwidth appear. Historically, MPLS was also born as an alternative forwarding architecture [26]. The idea of separating routing and forwarding is also present in other future networking proposals [27]. The concept of pathlets also has similarities to MPLS-like forwarding, though used in the inter-domain setting [28].

There are two relevant works connected to our proposal of pushing back the state to the CEs. The flow routing paper by Roberts [29] propose forwarding state for each flow. Still, the routing is done only once, and the flow state is added on the fly. Our work can be seen as a similar type of routing, but without the need of costly forwarding table entries inside the provider's network. A recent proposal, ICING [24] also ensures that the forwarding identifier constructed can only be used along its approved path.

VII. Future work and conclusions

In this paper, we have introduced MPSS, a stateless and flexible packet forwarding architecture. The data plane technology can be controlled by slightly extending (G)MPLS control plane protocols. When installing it as a forwarding plane for service provider networks offering Multicast VPNs, its quasi-statelessness is the key for scalability and maximizing revenue.

We will continue our work on MPSS architecture in several aspects. In the architecture front, one of our goals is to extend the architecture to multi-domain environments. One track of research is in extending the presented MPSS solution with virtual links, providing better performance by adding only a small amount of state. We further plan to explore the issues with forwarding plane security. Also, we would like to examine inter-operability issues with MPLS.

Our on-going efforts are to build a proof-of-concept prototype for the L3VPN use case, where the data plane functionalities of CE, PE and P routers are implemented in Click. The control plane features will be implemented by extending DRAGON [30].

Acknowledgment

This work was partially supported by the EU FP7 PSIRP project under grant INFSO-ICT-216173, and by TEKES as part of the Future Internet program of TIVIT (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT).

References

[1] J. Saltzer *et al.*, "On the naming and binding of network destinations," in *Local Computer Networks*, 1982, pp. 311–317.

[2] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," IETF, RFC 3031, Jan. 2001.

[3] P. Jokela, A. Zahemszky, C. Esteve, S. Arianfar, and P. Nikander, "LIPSIN: Line speed publish/subscribe inter-networking," in *SIGCOMM*, 2009.

[4] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas, "LDP specification," IETF, RFC 3036, Jan. 2001.

[5] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, "RSVP-TE: extensions to RSVP for LSP tunnels," IETF, RFC 3209, Dec. 2001.

[6] A. Farrel, J.-P. Vasseur, and J. Ash, "A path computation element (PCE)-based architecture," IETF, RFC 4655, Aug. 2006.

[7] A. Rosen and Y. Rekhter, "BGP/MPLS IP virtual private networks (VPNs)," IETF, RFC 4364, Aug. 2006.

[8] E. Rosen and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs," IETF, Internet-Draft draft-ietf-l3vpn-2547bis-mcast-09, nov 2009, work in progress.

[9] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[10] C. Esteve, P. Jokela, P. Nikander, M. Särelä, and J. Ylitalo, "Self-routing Denial-of-Service Resistant Capabilities using In-packet Bloom Filters," *Proceedings of European Conference on Computer Network Defence (EC2ND)*, 2009.

[11] A. Zahemszky and S. Arianfar, "Fast Reroute For Stateless Multicast," *The Workshop on Reliable Networks Design and Modeling RNDM 2009*, 2009.

[12] E. Karpilovsky, L. Breslau, A. Gerber, and S. Sen, "Multicast redux: a first look at enterprise multicast traffic," in *Proceedings of the 1st ACM workshop on Research on enterprise networking*. ACM, 2009, pp. 55–64.

[13] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Kaashoek, "The Click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[14] S. Yasukawa, A. Farrel, and O.Komolafe, "An analysis of scaling issues in mpls-te core networks," IETF, RFC 5439, Feb. 2009.

[15] S. Bhatnagar, S. Ganguly, and B. Nath, "Creating multipoint-to-point LSPs for traffic engineering," *IEEE Communications Magazine*, vol. 43, no. 1, pp. 95–100, 2005.

[16] D. Applegate and M. Thorup, "Load optimal MPLS routing with N+ M labels," in *IEEE INFOCOM*, vol. 1, 2003, pp. 555–565.

[17] A. Gupta, A. Kumar, and R. Rastogi, "Exploring the trade-off between label size and stack depth in MPLS routing," in *IEEE INFOCOM*, vol. 1, 2003, pp. 544–554.

[18] B. Yang and P. Mohapatra, "Edge router multicasting with MPLS traffic engineering," in *IEEE International Conference on Networks (ICON 02)*, 2002.

[19] F. Solano, R. Fabregat, Y. Donoso, and J. Marzo, "Asymmetric tunnels in P2MP LSPs as a label space reduction method," in *IEEE ICC*, 2005.

[20] F. Solano, R. Fabregat, and J. Marzo, "A fast algorithm based on the MPLS label stack for the label space reduction problem," *Proc. IEEE IP Operations and Management (IPOM 2005)*, 2005.

[21] F. Solano, T. Stidsen, R. Fabregat, and J. Marzo, "Label space reduction in MPLS networks: how much can a single stacked label do?" *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 6, pp. 1308–1320, 2008.

[22] G. Apostolopoulos and I. Ciurea, "Reducing the forwarding state requirements of point-to-multipoint trees using mpls multicast," *ISCC 2005 Proc*, pp. 713–18, 2005.

[23] I. Martinez-Yelmo, D. Larrabeiti, I. Soto, P. Pacyna *et al.*, "Multicast traffic aggregation in MPLS-based VPN networks," *IEEE Communications Magazine*, vol. 45, no. 10, p. 78, 2007.

[24] A. Seehra, J. Naous, M. Walfish, D. Mazieres, A. Nicolosi, and S. Shenker, "A policy framework for the future Internet," in *HOTNETS*, 2009.

[25] C. Metz, C. Barth, and C. Filsfils, "Beyond MPLS Less Is More," *IEEE Internet Computing*, pp. 72–76, 2007.

[26] P. Newman, T. Lyon, and G. Minshall, "Flow labelled IP: A connectionless approach to ATM," in *IEEE INFOCOM*, vol. 96, 1996.

[27] K. L. Calvert, J. Griffioen, and L. Poutievski, "Separating Routing and Forwarding: A Clean-Slate Network Layer Design," in *In proceedings of the Broadnets 2007 Conference*, September 2007.

[28] P. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 111–122, 2009.

[29] L. Roberts, "The Next Generation of IP-Flow Routing," in *SSGRR International Conference, LAquila, Italy*, 2003.

[30] T. Lehman, J. Sobieski, and B. Jabbari, "DRAGON: A framework for service provisioning in heterogeneous grid networks," *IEEE Communications Magazine*, vol. 44, no. 3, pp. 84–90, 2006.