Autonomicity in Virtual Private Network Provisioning for Enterprises

András Zahemszky
Ericsson Research, NomadicLab
Email: Andras.Zahemszky@ericsson.com

Petri Jokela
Ericsson Research, NomadicLab
Email: Petri.Jokela@ericsson.com

Tony Jokikyyny
Ericsson Research, NomadicLab
Email: Tony.Jokikyyny@ericsson.com

***Abstract*—Large enterprises usually require Virtual Private Network (VPN) services provisioned by the network operator. Also, there is an emerging need for supporting multicast communications, i.e. one host communicate with other hosts located in multiple remote sites. While MPLS-based IP VPNs are proven to be scalable, current approaches for extending it with multicast features involve potential state explosion, some bandwidth inefficiencies in the operator network or complex management tasks to find a good balance between forwarding state and bandwidth usage. These properties are direct consequences of the current MPLS and network-layer multicast forwarding approaches, as state should be maintained in the forwarding plane for each tree in each intermediate node.**

In this paper, we build on a stateless Bloom-filter-based forwarding plane installed in the service provider's network. By moving the state into the packet headers from the nodes, new trade-offs appear due to the probabilistic nature of Bloom filters. We highlight autonomic scenarios, such as self-configuration of addresses, resource management in the network, simple autonomic provisioning of dynamic multicast trees and self-optimization of forwarding performance.

I. Introduction

Autonomic computing is [1] a rather well established term in computer science. More recently, work has been started for extending the autonomic computing concepts in order to create complete self-managing systems to establish autonomicity in the networks. The motivation is clear and comes from both academia and industry. It includes complexity and cost of operation, or to provide the way forward to enable pervasive and ubiquitous communications. Becoming able to develop communication networks that can self-manage without the conscious effort of the operator or the user is the overarching vision that is believed to be achievable through the utilization of self-* properties [2].

For a system to be autonomic, it should have several self-* properties such as self-awareness, self-configuration, self-healing etc. The EFIPSANS research project is taking an evolutionary approach to implement these properties into existing and future systems through a Generic Autonomic Networking Architecture (GANA) [3]. As a main enabler the GANA uses control-loops for achieving full self-manageability of communications systems by design. The GANA control-loops can be applied at four different levels of abstraction from the lower protocol level, then the function level (eg. routing, mobility) up to the higher node and network levels. Figure 1 shows a model of a generic autonomic networked system and its associated GANA control-loop for implementing autonomicity. In the GANA model, Decision-making Element (DE) drive the control loops.

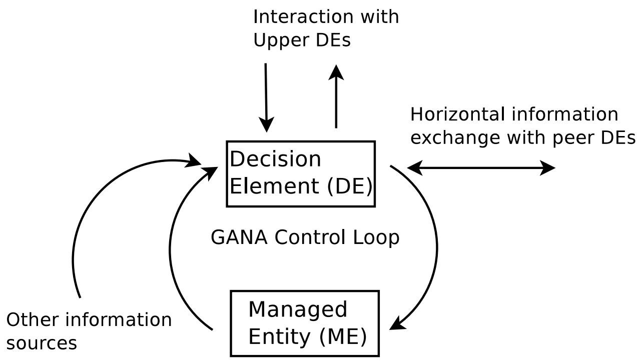
Interaction with Upper DEs
Horizontal information exchange with peer DEs
Decision Element (DE)
GANA Control Loop
Managed Entity (ME)
Other information sources

Fig. 1. The control loops in GANA

In a simplified view, the control loops work as the following. DEs are gathering information from the resources/Managed Entities (ME) they manage, from monitoring components, and also, from their peer and sibling DEs. Based on their view, they execute an autonomic behavior/select an algorithmic scheme/or a policy that they enforce on the Managed Entity. Now, the network state may change based on the actions, and that can trigger new behaviors in this DE, or any other DEs inside the node or in the network.

The focus of this paper is on an experiment to bring autonomicity into virtual network provisioning for enterprises. To realize this, we investigate some autonomic properties of legacy MPLS-based VPNs and a novel solution, which is utilizing a stateless, Bloom filter-based forwarding in the network, developed by another research project involving the authors. We apply the GANA-model according to requirements [4] to our architecture, called Multiprotocol Stateless Switching [5], to reveal its contributions to the design for autonomicity.

The rest of the paper is organized as follows. Section II introduces the technical background of this work. Section III shows autonomic behaviors in the VPN architecture. After overlooking the related work in Section IV, Section V concludes the paper.

II. Background

A. Layer 3 Virtual Private Networks

Sites in physically separated locations can be connected over operator's MPLS network as depicted in Figure 2. Customer VPN sites connect with Customer Edge (CE) routers to the Provider Edge (PE) routers. MPLS tunnels are set up between PEs, and operator's routers (P) forward traffic with MPLS labels [6].

The VPN routing information is exchanged between PEs using BGP. The PEs maintain the needed forwarding entries for the VPNs that they serve. Overlapping address spaces are supported in the architecture.

Unicast forwarding uses two stacked MPLS labels; the outer label is used for PE to PE forwarding, and the inner label for delivering the packet to the correct customer VPN at the egress PE. The inner MPLS labels are advertised using BGP.

When the customers require IP multicast communication between receivers in different sites, point-to-multipoint connections are needed.

The early, and the most deployed solution [7] utilizes PIM (Protocol Independent Multicast) for multicast routing information distribution, and GRE-based tunneling for data delivery. For each VPN, all multicast data packets are forwarded on the VPN-specific multicast tree, spanning all the PEs serving sites for the specific VPNs. If there are no receivers for a certain multicast group in a certain remote site, traffic is wasted. Therefore, the specification allows to set-up dedicated multicast trees for specific multicast groups in the core; this way, bandwidth is not wasted, but even more forwarding state appears in the Provider routers, which might negatively affect scalability.

In the Next Generation Multicast VPN architecture [8], the control plane is utilizing BGP, just as in the unicast case. For delivering data packets, the options include IP multicast, ingress replication (utilizing only unicast tunnels), MPLS point-to-multipoint LSPs signalled with RSVP or LDP. The trees can be shared between different VPNs, and can be used by several customer multicast groups. The trees can be selective trees, covering only a subset of PEs belonging to a VPN. The specifications offer a powerful set of tools to be able to fine-tune the trade-offs between bandwidth usage and state inside the operator's network. However, complex algorithms are used to decide that in which multicast tree the customer multicast groups are bound to, as the number of trees maintained in each node is limited. Consequently, tree aggregation techniques are used, with the expense of extra traffic to uninterested PEs.

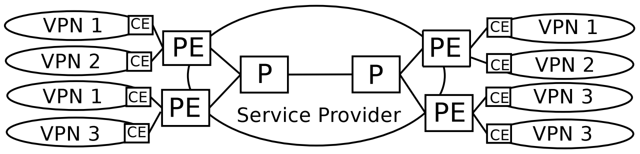

Fig. 2. L3VPN scenario

B. Forwarding with in-packet Bloom filters

As shown in our prior work [9], source routes and trees can be efficiently encoded into the packet header using Bloom filters [10] and locally naming links instead of nodes. Once the path or tree is determined, the forwarding identifier, called in-packet Bloom Filter (iBF), is formed by compressing the set of link identifiers into a Bloom filter by ORing them together (see Figure 3 a).

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. 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. A simple method in LIPSIN [9] is for each node to store one or more static link identifiers and then to choose the one used based on a value carried in the packet.

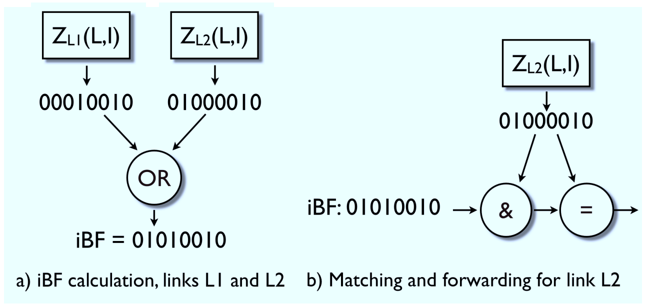

Fig. 3. Forwarding with in-packet Bloom Filters

The forwarding node makes a forwarding decision for each outgoing link o by checking if the iBF 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 (see Figure 3 b). If the iBF matches multiple outgoing Link IDs at the node, then the packet is forwarded to each of them, resulting in multicast. The BF matching process results sometimes in false positives. In iBF forwarding, this causes some additional packets to be delivered in the network, typically over one link. The probability of false positives increases when more links are added to an iBF. For performance and security reasons, the maximum number of bits set to one is limited to a certain percentage of total bits in iBF, e.g. to 50%.

In [11], it is shown that storing and using a key as the local input value for $Z(L, I)$, it is possible to achieve constant sized forwarding table and improve security. For example, the link identifier calculation can be based on the flow information (IP 5-tuple), quality-of-service bits, MPLS label, the incoming interface, the outgoing interface, and the actual key. This means that the iBF is only valid on its intended path, only for a specific time and quality of service, and only with the given 5-tuple.

C. *MPSS: Multiprotocol Stateless Switching*

MPSS (Multiprotocol Stateless Switching) [5] is a continuation of the work on the in-packet Bloom filter based forwarding layer. Shortly, MPSS targets those networks where currently MPLS is installed for reasons such as flexible management of traffic, rerouting around failures, or providing Layer 2 or Layer 3 VPN services, both unicast and multicast. The primary difference in the concept of the two systems that in MPSS, MPLS labels are replaced by small Bloom filters, encoding the path or the tree the packet needs to follow. Thus, in the default case, the state related to the Label Switched Paths (LSPs) are drastically reduced, as in MPSS, the iBF already holds sufficient forwarding information about the whole path or tree in the MPSS-enabled network.

To provide the sender with an iBF, the network needs to perform three or four steps: a) compute the path, b) determine an iBF corresponding to the path, c) optionally, reserve the resources on the forwarding nodes, and d) providing the sender with the iBF. These steps can each be performed separately, and each can be accomplished either in an off-path or in an on-path, hop-by-hop manner. The off-path solutions utilize a link state routing protocol such as OSPF-TE, IS-IS-TE, or other similar protocol, for distributing link state information. The on-path solutions rely on extending existing hop-by-hop protocols. Most often the path computation and iBF determination steps are combined, but it is possible to use, e.g., off-path path computation and on-path iBF collection. Another possibility is to combine resource reservation and iBF provisioning into a single step.

While we refer to [5] for further details on the architecture, we briefly sketch two scenarios to emphasize the flexibility of MPSS. In the first scenario, consider that the tree is requested by the source node with requirements such as bandwidth constraints from a remote Path Computation Element (PCE). The PCE computes the tree satisfying the constraints, and sends the information to the source. With the strict source routing information, the source initiates an RSVP-TE process, where the resources are reserved and the iBF is calculated hop-by-hop according to the forwarding decision (and based on some flow information, optionally, cf. as in zFormation [11]. In another scenario, the source node can compute the iBF, as OSPF-TE could be extended to advertise the link identifiers. Now, each node can compute the tree and even the iBF, and if resource reservation is not needed, the iBF can be immediately used for communication, without any additional signalling delay (cf. RSVP-TE explicit routes with zero bandwidth reservation, where the hop-by-hop path setup is still needed to configure the forwarding tables).

One promising application of MPSS is provisioning Multicast VPN services to organizations. As a forwarding solution in the service provider's network provisioning Multicast VPNs, MPSS has the promise of easing the trade-off and the difficult process of fine-tuning, as it offers stateless multicast, though with the penalty of false positives, i.e. a controllable amount of unnecessary packet forwardings due to probabilistic reasons.

III. Autonomic behaviors in the VPN architecture

In this section, we show the application of GANA to our in-packet Bloom filter based forwarding architecture, and the L3VPN architecture, offered by MPLS or MPSS-enabled networks. We also show that how some of our previous design choices can be implemented in autonomic manner during the network operation.

A. *Self-configuration of forwarding table entries*

The iBF-based forwarding utilizes source routing. The link identifiers do not need to be centrally allocated in this system. Each node can select a random bitstring, which is long enough to avoid collision within the network (eg. 256 bits) and can determine the bit positions where the link identifier is set to one by applying k independent hashing functions with output values $1..m$ to it. The resulting link identifier is an m-bit long bitstring with k bits set to one. Link identifiers are uni-directional, which means that no coordination is needed with the neighboring nodes apart from a neighbor discovery procedure (note that capability description and negotiation might be still needed in some scenarios).

The determined link identifiers are then advertised with a link state routing protocol towards all nodes or to a central entity. The self-configuration of the local "addresses" and the distribution of them via a routing protocol make it possible to determine and use MPLS-like source routing paths in the network without any additional signalling. If bandwidth reservation is required, as in the RSVP-TE case for MPLS, we envision a bandwidth broker-like Decision Element (see III-C).

In the dynamic case, routers have a secret key, which they either share securely with entities computing iBFs (possibly few elements only), or the iBF-calculation should be done hop-by-hop with a signalling message, giving up some benefits of the favorable properties of self-configuration.

B. *Auto-configuration of VPNs*

A relevant security threat in any VPN architecture is the so-called cross-connectivity threat, which means that traffic from one VPN leaks into another VPN, due to misconfiguration or physically connecting the CE to the wrong "slot".

Misconfiguration errors can be better avoided, if manual intervention is restricted. In the case of a VPN network, this means using a central management system, where authentication of new CEs are performed and the configuration is automatically sent to the PE router. Basically, the new configuration information needed for the router is the Route Target (RT) attributes for BGP advertisements, and the new set of incoming and outgoing RTs.

When a new CE is connected to the PE, it authenticates itself to the Network Level VPN Management DE (or possibly to the Security Management DE). After the authentication phase, the VPN Management DE informs the Routing DE in the PE node about the configuration it should apply, and finally the BGP parameters are set in the node via the appropriate Managed Entity.

Also, in the unicast case, PE-to-PE tunnels has to be built, if the newly connected site belongs to a VPN the PE has not been connected before. If the MPLS tunnels are built with LDP, then the tunnel set up will be immediately started, as it should be initiated by the egress PE. In the case of RSVP, the set up should start from the ingress node - the state-of-the-art solution is automesh [12], where PEs advertise their existence with OSPF-TE Opaque LSAs, so others can initiate LSP set-up after receiving the information.

Because of the different nature of MPSS, the automesh solution might not be needed. By only getting the BGP route updates, the ingress PEs can itself compute or ask a remote entity to calculate an iBF to the newly connected egress PE. Similar considerations are applicable to the multicast case: by knowing the receiver PEs, an appropriate representation of the multicast tree can be computed locally or remotely, without the need of configuring the intermediate nodes.

C. *Resource-aware routing*

By definition, Traffic Engineering has the task of improving the network's resource utilization by controlling what paths the traffic takes. TE as such is one of the most important features of MPLS/GMPLS networks.

In MPLS, traffic engineered LSPs are set up with RSVP-TE. Usually, constraints are given for the path and a candidate path is calculated. Finally, the new LSP is signalled hop-by-hop, meaning that the forwarding tables are set, and the resources are reserved in the control plane.

Restricting to only intra-domain operation, the process can be described by applying the GANA model and using the iBF-based forwarding. We assume a Network Level Resource Management DE, which is aware of all resource reservations in the network. In the Provider Edges, Node Level Resource Management DEs (RM DEs) operate. The Node Level RM DEs see the resource situation in the network by periodically synchronizing with the Network Level RM DE. When a path is requested with some constraints (usually by the source), they respond with a suitable path to the co-located iBF Provisioning DE, and inform the Network Level RM DE how it modified the resource situation in the network. This ensures that the Network Level RM DE always sees the current resource reservations. When the iBF is calculated by the iBF Provisioning DE, it is installed into the forwarding table by the Function Level Routing Management DE at the source.

In some scenarios, there is no Node Level Resource Management DE, and the path selection and the iBF calculation (by the iBF provisioning DE) is performed remotely, in a central entity (cf. the PCE model in GMPLS networks).

The Resource Management DEs can be stateful. By keeping track the resource reservations for each path, they can modify the path and thus reorganize the resource situation in the network, if needed. This is the consequence of the fact that the reservation requests arrive one-by-one, and can leave dynamically and in many cases better resource allocation can be achieved, if the arrival and leaving "history" is not considered.

Finally, in a multi-domain network, we envision a (G)-MPLS-like multi-domain solution, where a set of Path Computation Elements (PCEs) are responsible for computing the whole path, by each computing a separate segment.

D. *Autonomic provisioning of dynamic multicast trees*

In a dynamic environment, the membership of multicast groups constantly change. In many cases, the multicast trees in the operator's network need not to be adjusted: for example, a new receiver joins but the site already has receivers for the group.

However, if the new receiver is the first in the site joining the group, the multicast tree in the service provider might be changed in the MPLS case, and the iBF must be modified in the MPSS solution.

Consider that the multicast tree in the operator that carries the customer's multicast group G originates from PE_1, and covers PE_2, PE_3. Now, a new receiver, behind PE_4 joins the multicast group.

We select one scenario from the options of L3VPN specifications, where the packets of G are carried by an RSVP-signalled point-to-multipoint LSP T, and other customer multicast groups, some even belonging to different VPNs are transported on T. Now, when PE_1 is informed via BGP that the multicast group has a new receiver, it has to make a decision. If there exists a tree T_2, that cover exactly the receivers of G, then the group is put to that tree. If it is not available, multiple options exist. One option is that it extends T by adding PE_4 with the appropriate RSVP signalling sequence. This means that the other multicast group's packets will be transported to PE_4, by wasting some bandwidth in the core network.

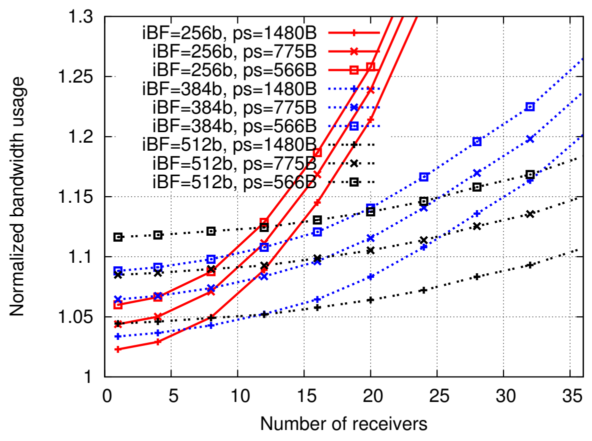

Fig. 4. Bandwidth overhead of iBF-forwarding for different packet sizes and iBF-sizes

Another option is that it binds G to another existing tree T_3, which covers PE_1, PE_2, PE_3, PE_4, and some other PEs. The third option that it initiates the creation of a new tree T_4, covering the receivers of G.

The decision of the ingress node is based on the policy and the following viewpoints: how much bandwidth is wasted with the different options? What is the utility of the new tree? For example, if G carries low-bandwidth traffic, creating T_4 for the sake of G is not beneficial, as the state maintenance require additional complexity. In this case, it might be a good decision to bind G to T_3, even if some PEs will get the packets unnecessary.

When utilizing MPSS in the service provider's network, the decision to be made in the ingress node is simpler. The ingress node has to acquire an iBF, that describes a tree, covering PE_4as well. This can be achieved by requesting from local or remote DEs an iBF describing the path from PE_1 to PE_4, which is ORed to the iBF used earlier. If advanced traffic engineering is applied, the operations are similar as it was described shortly in III-C.

E. Self-optimization of forwarding performance

Due to false positives in Bloom filters, the iBF-based forwarding has a certain overhead, increasing for larger trees. The forwarding overhead (the ratio of total sent bytes to the useful traffic, including per-packet overhead) for different Bloom filter sizes can be found for the AS3257 topology (161 nodes, 328 links) on Fig. 4.

We describe how the network can autonomously react to increased amount of false positives for certain multicast trees. First, the network has to find which Bloom filter causes too much extra overhead. This can be determined by seeing that the fill factor (the ratio of 1s in the iBF) is above a threshold, say 30-40%. Additionally, the ingress node (the iBF Provisioning DE) periodically can check that exactly how much false positives happen - this can be done by following the packet's route on the graph of the network.

By interaction of the iBF-provisioning DE and the Resource Management DE, the decision can be made to split the Bloom filter into two, ie. serving the receivers of the multicast group with two trees; one set is served by the first tree, while the other is with the second. This means that the unnecessary forwardings due to false positives decrease, but some copies might traverse the same link multiple times (if it is impossible to create to disjunct trees from the original one). This depends not only the amount of false positives, but also the extra traffic caused by the group; the benefit of applying this optimization for a high-bandwidth group is much higher than for a group producing a few kbit/s traffic.

Another possible operation is to increase the size of the Bloom filter. While this means that the per-packet overhead will be larger, the number of false positives, thus the amount of extra forwardings decrease. This operation can be for example implemented by allowing a set of "link identities" for a single node, which are of different sizes, and can be referenced by an index (a modification of an idea found in [9], [13], where each link identity is of the same size). The packets, apart from the iBF, contain also the index telling the node which link identity's presence to check. When the link identifiers are calculated dynamically for each packet, this index will be one parameter for the link identifier calculation algorithm. The decision to increase the size of the iBF should be performed in concert of several DEs, and is based on the characteristics of the multicast group - bandwidth, typical packet sizes (could be provided by a so-called Monitoring DE with the task of collecting statistics about the network usage), the fill factor of the iBF (can be retrieved from the iBF-provisioning DE) and the overall resource situation in the network (Resource Management DE). As an example, a low bandwidth multicast group experiencing too many false positives may not waste as much bandwidth as a high bandwidth multicast group sending packets over one or two extra links. In the latter case, the length of the iBF should be increased.

So far, the forwarding fabric is quasi-stateless, and now we present how the overall performance can be increased by adding a small amount of state into the forwarding nodes. LIPSIN introduced the notion of virtual links; a set of links can be referenced by a single link identifier. While the new link identifier has to appear in the forwarding table of each node on the virtual link, the fill factor of the iBF is reduced as only one link identifier is inserted instead of several ones. When the decision is made to create a virtual link, the Function Level Forwarding DE needs to be notified to add the state into the forwarding table.

Another virtual link-style optimization technique is to add a dedicated forwarding state into each intermediate node for the multicast group. A bit in the forwarding header can indicate whether a tree identifier or an iBF is present in the packet header. This can be beneficial for medium- and large multicast groups, with medium- to large bandwidth demands.

IV. Related work

Basically, our approach is a new framework for fine-tuning the difficult and intrinsic trade-off of bandwidth usage and amount of forwarding state with respect to multicast trees in MPLS VPNs.

As for MPLS multicast, Edge Router Multicasting [14] and the scheme proposed by Boudani et al [15] are techniques to build point-to-multipoint LSPs by utilizing only unicast LSPs in the network. In these solutions, the amount of links a multicast trees contain is not optimal.

Focusing also on the state usage of MPLS multicast trees, Solano et. al propose the usage of asymmetric tunnels [16] for reducing the label space usage. In their most recent work, they reduce per-packet overhead by combining the concept of tunnels with label merging [17], requiring a single label per packet.

The authors of [18] present an approach, where routers can cooperatively decide on aggregating labels. The work in [19] summarizes the mechanisms of traffic aggregation in MPLS-based VPNs, and presents a model for bandwidth saving and forwarding state in the scenario, where multiple VPNs share a single tree in the provider's network.

Karpilovsky et al [20] present a thorough analysis of multicast traffic in an operator offering Multicast VPN services to customers. Their conclusion is that the multicast trees in the service provider's network are sparse, covering around 2-3 receivers, and only 13.3% has 20 receivers in average.

V. Conclusions

In this paper, we discussed several autonomicity questions in different enterprise VPN scenarios: supporting unicast and/or multicast communications between the sites, utilizing legacy MPLS LSPs and trees, or utilizing in-packet Bloom filters (iBF) for the forwarding plane.

We discussed, that in certain scenarios, the stateless nature of the in-packet Bloom filter-based forwarding can ease the management of the network, and provides a first step towards autonomicity. We applied the GANA model to our iBF-based forwarding architecture, and by introducing new Decision Elements, we illustrated autonomic behaviors for optimizing the network usage, by allowing trading-off the per-packet overhead, amount of forwarding state, and bandwidth usage in the network. Concrete optimization algorithms are subject of our future work.

Acknowledgment

This work was sponsored by the EU FP7 EFIPSANS project under grant INFSO-ICT-215549 and by the EU FP7 PSIRP project under grant INFSO-ICT-216173.

References

[1] R. Sterritt, "Towards autonomic computing: effective event management," in *27th Annual NASA Goddard/IEEE Software Engineering Workshop, 2002. Proceedings*, 2002, pp. 40–47.

[2] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[3] EFIPSANS project, "Third draft of autonomic behaviours specifications (ABs) for selected diverse networking environments," EFIPSANS project, Deliverable D1.5, 2009.

[4] ——, "Third draft of requirements specifications (RQs)," EFIPSANS project, Deliverable D1.6, 2009.

[5] A. Zahemszky, P. Jokela, M. S. S. Ruponen, J. Kempf, and P. Nikander, "MPSS: Multiprotocol stateless switching," in *13th IEEE Global Internet Symposium*, 2010.

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

[7] E. Rosen, Y. Ciai, and I. Wijnands, "Cisco Systems' Solution for Multicast in MPLS/BGP IP VPNs," IETF, Internet-Draft draft-rosen-vpn-mcast-15, jun 2010, work in progress.

[8] E. Rosen and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs," IETF, Internet-Draft draft-ietf-l3vpn-2547bis-mcast-10, jan 2010, work in progress.

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

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

[11] 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.

[12] JP. Vasseur (Ed.), JL. Leroux (Ed.), S. Yasukawa, S. Previdi, P. Psenak, and P. Mabbey, "Routing Extensions for Discovery of Multiprotocol (MPLS) Label Switch Router (LSR) Traffic Engineering (TE) Mesh Membership ," IETF, RFC 4972, Jul. 2007.

[13] A. Zahemszky, A. Császár, P. Nikander, and C. Esteve, "Exploring the pubsub routing/forwarding space," in *International Workshop on the Network of the Future*, 2009.

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

[15] A. Boudani, B. Cousin, and J. Bonnin, "MPLS Multicast Traffic Engineering," *IEEE ROC&C*, 2003.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.