Forwarding of Multicast Packets with Hybrid Methods Based on Bloom Filters and Shared Trees in MPLS Networks

Gonzalo Fernández-Del-Carpio
School of Electronics and Communications Engineering
Universidad Católica San Pablo
Arequipa, Peru
Email: gfernandezdc@ucsp.edu.pe

David Larrabeiti and Manuel Urueña
Dept. of Telematics Engineering
Universidad Carlos III de Madrid
Madrid, Spain
Email: {dlarra, muruenya}@it.uc3m.es

***Abstract*—Multicast forwarding in the context of IP-MPLS networks for services like VPLS has important scalability issues. Alternatives to per-VPLS tree construction based on aggregation of multipoint requests onto shared trees have been studied. Other works have focused on stateless forwarding based on Bloom filters. Both suffer from some type of forwarding anomalies. This paper proposes the combination of both forwarding modes: shared multicast trees and an efficient variant of stateless switching based on Bloom filters. Simulations prove that constraining the forwarding to the shared tree is an effective mechanism to prevent most side effects of Bloom filter-based forwarding, making the combined technique a good trade-off in terms of forwarding state consumption, yielding best performance in terms of overhead with respect to each technique in isolation.**

I. Introduction and Related Work

Making use of multipoint LSP (Label Switched Path) support in a multi-service IP-MPLS network to take advantage of the benefits of multicast in terms of transmission efficiency, entails also the well-known scalability problem of multicast over any type of switching technology. This problem consists of the fact that the most efficient solution in terms of bandwidth consumption -creating a tree per source and group- is also the most network resource consuming in terms of forwarding state.

The creation of a shared tree (ST) per-group, like PIM-SM (Protocol Independent Multicast - Sparse Mode) does in IP routing, improves the situation but does not solve the problem completely, as the number of groups (multicast requests) is unbounded. Furthermore, in the case of services like VPLS (Virtual Private LAN Service) where the operator emulates the Broadcast/Multicast/Unknown service of Ethernet for their clients, the use of a tree per VPLS breaks a fundamental rule of MPLS VPN service provision architectures: VPNs can add state to the Provider Edge (PE) routers where VPN sites are attached to (see Fig. 1), but, by no means, any VPN can add state to the core nodes. Hence, the standard solution consists of *ingress replication*, where the ingress PE copies the broadcast/multicast frames over as many point-to-point LSPs as PEs with sites attached the Virtual Private LAN has.

Given that ingress replication does not take full advantage of multipoint, several solutions have been devised by researchers. One alternative in the case of MPLS is the aggregation of multipoint requests onto a fixed number of shared trees [1]. The amount of state required in the core is determined by the chosen number of trees. The main drawback of this technique is the fact that packets are delivered to all the PE leaves of the shared tree, even if no site of the VPLS is served by those PEs. This technique can be improved by heuristics that try to put together the most similar requests [2].

Another alternative is *stateless switching* based on *Bloom Filters* (BF) [3]. Stateless switching is a kind of efficient source routing, whereby the path or tree of node interfaces is coded into the header of the packet as a BF, and packets are copied to all the interfaces that match the BF. This approach has two interesting advantages: (a) no forwarding state is required, and (b) the paths followed by packets can be optimal. However BF-based stateless switching also has drawbacks: (a) BF false positives create unnecessary forwarding branches, (b) BF false positives create unnecessary forwarding branches that may cause a loop in the tree, leading to packet storms. A lot of research work has been devoted to prevent or diminish these forwarding anomalies [4] but the probabilistic nature of BFs make all solutions imperfect. A few flows are sent to unintended PEs, the same problem as with shared trees, causing overhead transmission and switching. If instead of coding the interfaces with a BF, we encode only the addresses of the set of egress PEs (Destination Oriented Multicast vs. Tree Oriented Multicast [5]), the BF can be smaller, at the price of some per-egress state, but the forwarding anomalies due to false positives persist and IP trace-back check mechanisms are required to get rid of them.

In 2015, a new working group (WG) named *Bit Index Explicit Replication* (BIER) was chartered at IETF, aiming to define a deterministic form of multicast by coding nodes in a more explicit way than a BF. Given the required changes in the data plane its work is progressing as Experimental. The goal of BIER is to provide optimal multicast packet forwarding through a *multicast domain* without intervention of an explicit

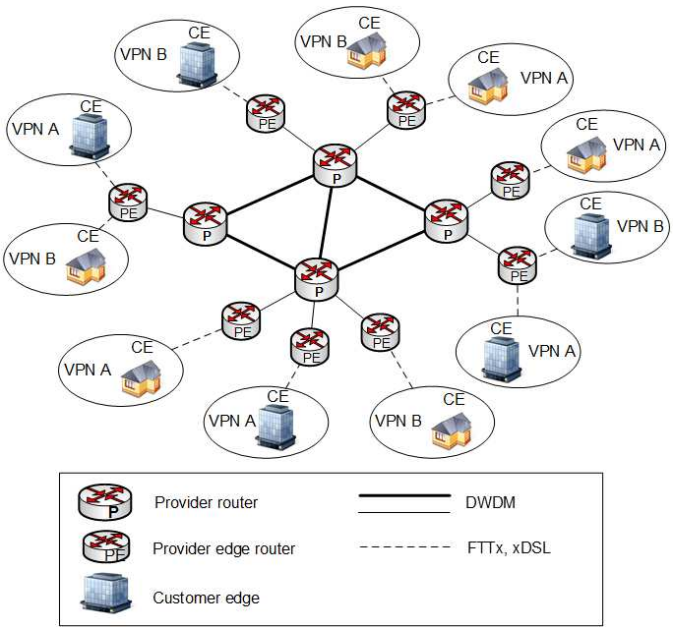


Fig. 1. MPLS VPN-based network scenario

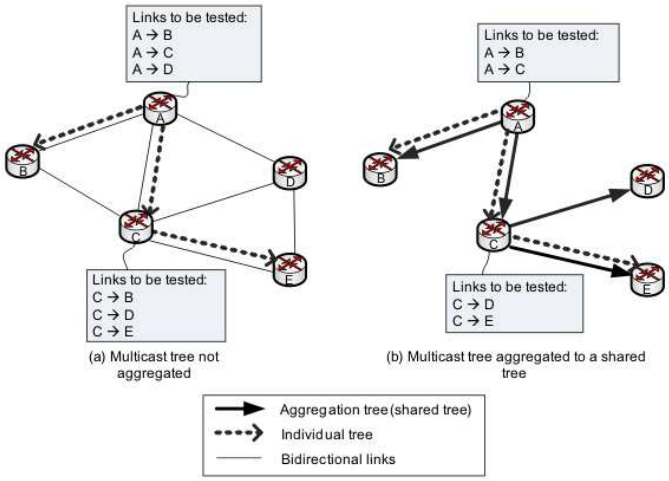


Fig. 2. Number of links to be tested with any Bloom filter-based technique without/with aggregation.

tree-building protocol, nor requiring intermediate nodes to maintain any per-flow state. In BIER, when a multicast data packet enters the domain, the ingress router determines the set of egress routers to which the packet has to be sent. The ingress router then encapsulates the packet in a BIER header. The BIER header contains a bit-string where each bit represents exactly one egress router of the domain. To forward the packet to a given set of egress routers, the bits corresponding to those routers are set in the BIER header. Packets follow the unicast shortest path tree to the destinations according to the bit-string. The advantages of BIER are that the tree is optimal, causes no per-flow state, only per-egress state. However, BIER also has scalability issues for big domains as the maximum number of egresses is limited by the bit-string length. A practical future-looking target was set to 256. Therefore BF still keeps on being a choice to study for big networks not partitioned into small BGP domains.

In this paper we focus on the large domain scenario and propose a combined shared-tree-BF mechanism to prevent forwarding anomalies; in particular BF loops are completely removed and most useless forwarding is eliminated (we call this latter anomaly *overhead traffic* in the rest of the paper). The proposed mechanism works as follows. Firstly, packets are sent to the root node of a shared tree in the core using an existing point-to-point LSP. The packet carries a header with a BF representing the tree of interfaces to use down the shared tree. At the root node, the packets are added a new label to travel down the shared tree. Both MPLS label and BF are employed to forward the packet: (a) an efficient variant of stateless switching [6] called D-MPSS [7] based on Bloom filters is applied (section II) to the packet to determine the output interfaces for that packet, but (b) only the shared tree's output interfaces for its input label are checked for membership. In [8] a hardware deployment on NetFPGA card of both stateless routing mechanisms was succesfully achieved, in order to prove their viability and to make a proposal to implement them.

The paper is structured as follows. Section II addresses the main concept of combinig BFs and shared trees, and III describes the approach in detail. Section IV gives simulation results and V presents conclusions.

II. Combining BFs and Shared Trees

We explore the possibility of realizing multicast forwarding by combining aggregation techniques with the use of Bloom filters. This new method is based on setting up two fields in the packet header: one corresponding to the MPLS label corresponding to the shared tree, and the other to the Bloom filter. Our proposal is mainly supported by the idea that the reduction in number of outgoing links to be evaluated, at each node, will reduce the overhead traffic generated by false positives. In the example presented in Fig. 2, picture (a) has an individual tree to forward packets from A to B, C and E. In the picture (b) the same tree is aggregated to a given shared tree (note that the individual tree is a subset of the ST). Observe that in the first case (without aggregation) the number of links to be checked (done with any of the Bloom filter-based techniques) in nodes A and C, are 6 in total; but in the second case there are only 4. As a consecuence, the ST limits the scope of the BF-based forwarding, and conversely, even though we are re-using the same ST for multiple multicast sessions, the multicast distribution over the ST is limited by the BF. Therefore they work as complementary mechanisms to fight overhead traffic.

Although this approach may be used for any type of network that requires to enable its provision of multicast services, we develop it for the scenario of an MPLS VPN-based network, as explained in [2]. In that scheme, u multicast VPNs (MVPNs) are aggregated to s shared trees, which are set up among the

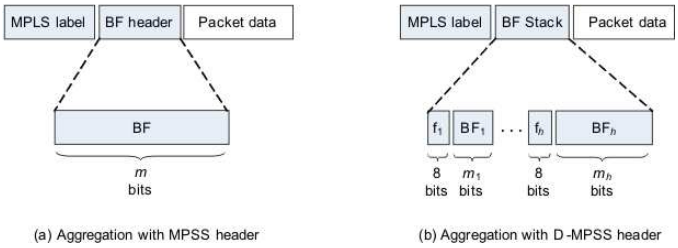


Fig. 3. Packet header of the technique proposed, using MPSS (a) and D-MPSS (b) as Bloom filter-based methods.

transport network. Core intermediate nodes keep information only for the forwarding of the STs, but do not have any knowledge about the VPN specific information of the packets that are traversing them. The VPN specific information is encapsulated in the packets at the PE source node, and is only inspected at the final destination PEs. The main difference here is that the Bloom filter is located between the MPLS label (corresponding to the shared tree) and the packet data, as shown in Fig. 3. The figure also outlines the difference between the two BF-based forwarding mechanisms used in this paper. MPSS uses a single BF to code the set of output interfaces of the tree for that packet [6]; each interface is assumed to have a unique identifier in the network. D-MPSS [7] has several BFs of growing size, one per tree depth, achieving a lower degree of forwarding anomalies than MPSS.

A way of understanding the power of constraining the bloomcasting to a shared tree can be obtained by analysing broadcast storms. In [9], [10] and [11] security vulnerabilities of the Bloom filter-based forwarding techniques were studied. In the attack of the 100% fill factor, an attacker sends a packet with a Bloom filter with all the bits set to 1; thus, the packet would match positively with every outgoing link ID, causing a high rate of false positives and a chain reaction. In the case of MPSS [6] this would generate a broadcast storm (Fig. 4.(a)), only limited by the packet's TTL. A simple protection to this attack is limiting the fill factor in the network and filtering out packets over e.g. 50%, at the price of potentially needing larger BFs to keep the fill factor under that rate. With the use of shared trees it would be possible to remove this fill factor limitation, as BF-matching is constrained to a loop-less tree. In the event of an injection of 100% fill factor packets, the generated packet flow would only spread until the shared tree bounds (Fig. 4.(b)).

III. The Hybrid Methods Proposed

A. Packet Forwarding

In our proposed mechanisms every node maintains two forwarding tables: the standard MPLS forwarding table, with the corresponding entries for shared tree forwarding; and the link IDs table, with the entries of every outgoing interface encoded as Bloom filters. When a packet arrives at a core node, it first makes the look up process of the header within the MPLS forwarding table; if it succeeds, then it performs the Bloom filter header membership check with the corresponding output links IDs table. Depending on what filtering technique is being used, the matching process follows the steps described in [6] for MPSS (with permutation, as in [4]) or D-MPSS [7].

Let us look at the example provided in Fig. 5, that represents the state information of nodes A and C. The two shared trees, ST-1 and ST-2, are correctly configured with their respective forwarding entries at the MPLS forwarding tables. To serve a given multicast VPN, the individual multicast tree of the figure has been aggregated to ST-1. The fact that the multicast tree is aggregated to ST-1 means that all the traffic of this tree would use the state information assigned to ST-1 (MPLS forwarding table). Since more individual multicast trees will be aggregated to ST-1 and ST-2, the inner Bloom filter of the packet will have to be matched according to the BF technique and the linkIDs table.

Following the same example of Fig. 5, in Fig. 6 two packets of the multicast VPN with its corresponding MPLS-BF headers are launched to the network from node A (for this example we assume that the BF technique adopted is MPSS). The multicast tree of the VPN demand should visit nodes B, C and E. By following the first check (the MPLS look up), according to the state information of the STs at nodes A and C (Fig. 5), packets should be delivered to nodes B, C, D and E. After the second check (the BF matching) the packet in Fig. 6.(a) is filtered adequately and it is only delivered to nodes A and E. However, the packet in Fig. 6.(b) gives a false positive and is delivered also to node D.

B. Two Models of Shared Trees to be Combined with BFs

1) First Model: Fit Shared Trees (FST): In this model, shared trees are used in the way it was introduced in [2], in which s shared trees are created for aggregating u multicast VPNs. By following this technique, the multicast VPNs are aggregated to the most similar available ST (that is why we call them *fit shared trees* (FSTs). If necessary, after the aggregation, the chosen ST can be extended in order to contain a new VPN. In this method, every ST has its own *rendezvous point* (RP) node (the root of the ST) and all the source nodes

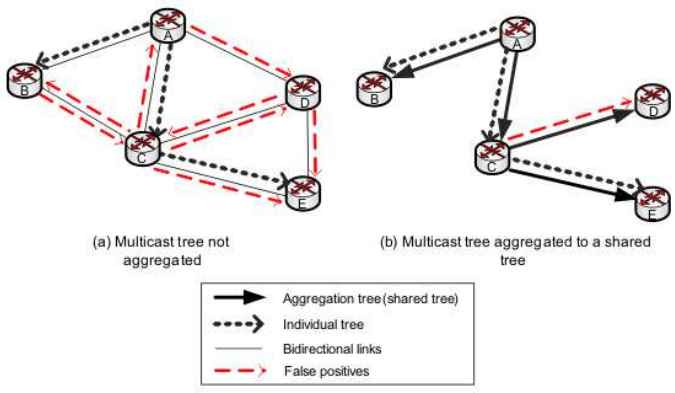


Fig. 4. Packet storm caused by a fully set Bloom filter with BF-forwarding (a) and how it is prevented with a shared tree (b).

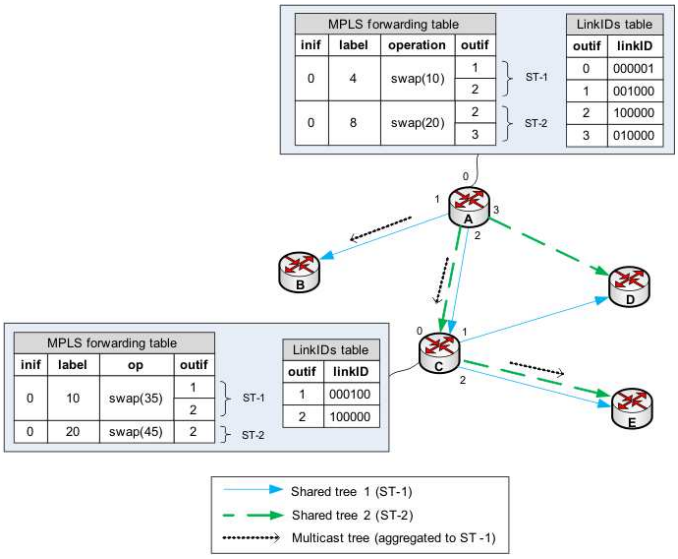


Fig. 5. Example of MPLS and link ID forwarding tables.

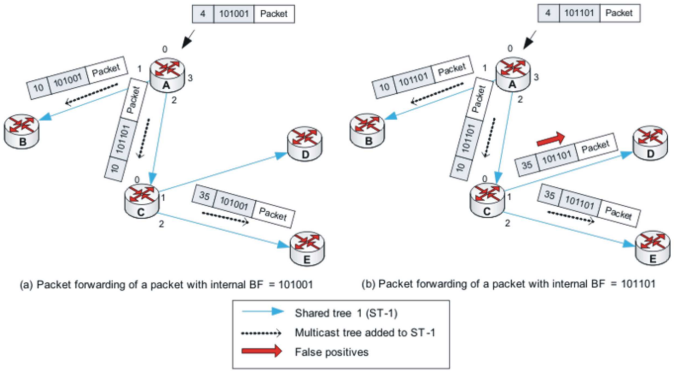


Fig. 6. Packet forwarding example using the hybrid technique.

have to deliver the packet first to it encapsulated in a LSP, and then the RP initiates the multicast forwarding process.

This model presents two drawbacks: (1) The creation of FSTs and *on-the-fly* computing for the accommodation of multicast demands into them is not a trivial task. (2) Although the RP of a FST is supposed to be the node with the minimum average distance -in number of hops- to the rest of nodes that take part of the VPN, in addition to the bandwidth wasted because of packets delivered to non-destination nodes, a number of transmissions will be wasted to get the RP.

$$\#entries_{FSTs}(node_i) \leq s + degree(node_i) \qquad (1)$$

2) Second Model: Broadcast Shared Trees (BST): Getting rid of the complexity and dynamic nature of FSTs, in this model a number of broadcast trees are created with diverse roots and sending PEs try to send the multicast traffic to the closest one. Taking the same MPLS-VPN network model of 1 and the provider aggregation (PA) model [12], PA nodes (nodes located between core and PE nodes) are selected as the roots of the broadcast trees, and each PA node acts as the

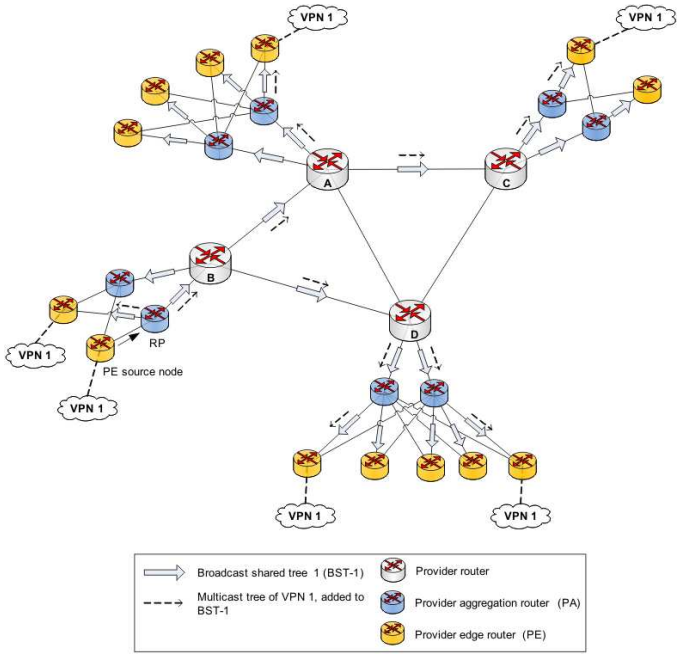


Fig. 7. A broadcast shared tree (BST) in an MPLS-VPN network.

rendezvous point of a BST. As shown in Fig. 7, note that the number of state entries at each node is exactly given by

$$\#entries_{BSTs}(node_i) = |PAnodes| + degree(node_i) \qquad (2)$$

Given that $|PAnodes|$ is a constant value, the number of entries in this model remains also constant at each node and, depending on the network size, in general $\#entries_{BSTs}(node_i) \leq \#entries_{FSTs}(node_i)$, except for high values of aggregation ratio AR (e.g. $AR = 100\%$), when a very few STs are built (e.g. one ST for aggregating all the demands). If state consumption is definitely an issue in the design, a single *spanning tree* can do the job, leading to

$$\#entries_{BSTs}(node_i) = degree(node_i) \qquad (3)$$

In Fig. 7 we present an example of a BST, which is supposed to be one of the 8 BSTs built by default before any transmission is done (each per PA node). As it may be observed, VPN 1 is attached to 6 of the points of presence or PE nodes, and all of them can potentially be the root of another multicast transmission. Let us suppose that the PE node labeled as *PE source node* originates a multicast transmission, in which case it chooses to be aggregated -among the 6 BSTs available- to the BST with the nearest RP (in this case, BST-1 with its RP attached to the provider backbone node B). As BST-1 has been set up to reach every node in the network, multicast packets will be delivered through it to all PE nodes. When any other site of VPN 1 has to send packets, the multicast transmission has to be aggregated to the BST with the nearest RP.

C. Performance Evaluation

This proposal, as being a hybrid technique that combines aggregation with Bloom filters for the multicast packet forwarding, is expected to reduce the overhead traffic generated

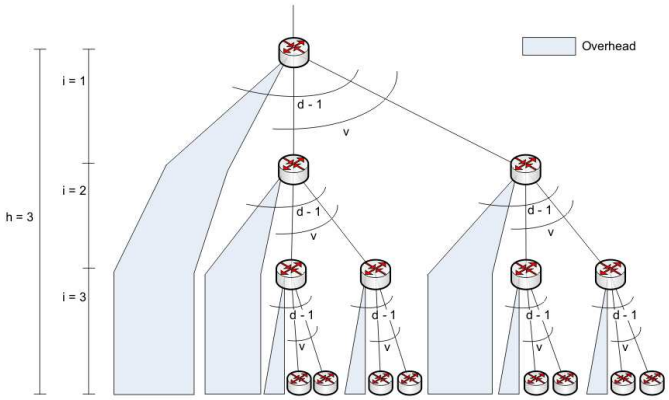


Fig. 8. Overhead propagation of MPSS in a regular network, with $d \geq 4$, $v = 2$ and $h = 3$.

by false positives, overcoming in this aspect -to the best of our knowledge- all the previous and current approaches that aim with the same problem. Nevertheless, as explained before, the counteract is that it needs a little of state information (specially for the BST model) and some extra processing. In this subsection we formulate the corresponding numerical analysis in order to measure these facts.

1) False Positives and Overhead: Both in MPSS and D-MPSS techniques, the *false positives rate* (fpr) depends on m, n and k values (see the correspondent equations 4 and 5), i.e. the Bloom filter size, the number of links/elements to be added to the BF, and the number of hash functions, respectively. In this sense, the fpr of the hybrid technique has the same formulation as the BF technique used in combination with the aggregation method. In equation 5 M is the upper size that each individual BF of size m_i (inside the stack of BFs) should have.

$$fpr = \rho^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-kn/m}\right)^k \quad (4)$$

$$fpr_i = \rho_i^k = \left(1 - \left(\left(1 - \frac{1}{M}\right)^{kn_i}\right)^{\frac{M}{m_i}}\right)^k \quad (5)$$

However, the real benefits of the hybrid technique should be seen in the reduction of the overhead traffic. Since now the domain of links for the BF matching process at each node is reduced to the set of branches of the FST or BST, less overhead packets are generated. Computing how many links are part of an ST might be an impossible task to be realized analytically in the case of the FST model, because it depends on the number of STs built, in the first place (note that the more STs are built, the less number of links they have). It also depends on the distribution of the multicast VPNs sites and the similitude among the multicast VPN demands.

However, the BST model is feasible to be modelled analytically. Let us consider a regular network (Fig. 8). As shown in [7], the overhead in MPSS is given by

$$oh(i)_{MPSS} = (d-v-1)\cdot fpr \cdot \frac{1 - ((d-1)\cdot fpr)^{h-i+1}}{1 - (d-1)\cdot fpr} \quad (6)$$

Where d is the constant degree of each node, v the number of branches of the tree at each node, h the depth of the tree, and i the depth of a node in the tree. In Fig. 8 an example of the behavior of the overhead propagation in such a regular network is shown. The shortest BST from the RP node will have $N-1$ edges, where N is the total number of nodes of the network. Given that N is a value dependent on the network topology, and impossible to derivate from the other network variables, we consider that at the i-th level of the tree there are $\delta \cdot v^{i-1}$ actual nodes. Let us consider δ as the scalar value that defines the network dimensions, so that

$$N = \delta \cdot n \quad (7)$$

where n (the constant number of branches of the tree) is

$$n = \sum_{i=1}^{h} v^{i-1} \quad (8)$$

$$n = v \cdot \frac{(1 - v^h)}{(1 - v)} \quad (9)$$

Now the average fanout (the average number of branches of the BST at any node of the i-th level) is

$$fanout(i)_{BST} = \frac{\delta \cdot v}{(d-1)^{i-1}} \quad (10)$$

Thus, following the formulation from Eq. 6, when a multicast tree is encoded as an MPSS Bloom filter and is aggregated to the BST described before, the total number of consecutive false positives generated is

$$totalOvh_{BST-MPSS} = \sum_{i=1}^{h} \left(fpr^i \cdot \prod_{j=1}^{i} fanout(j)_{BST} \right) \quad (11)$$

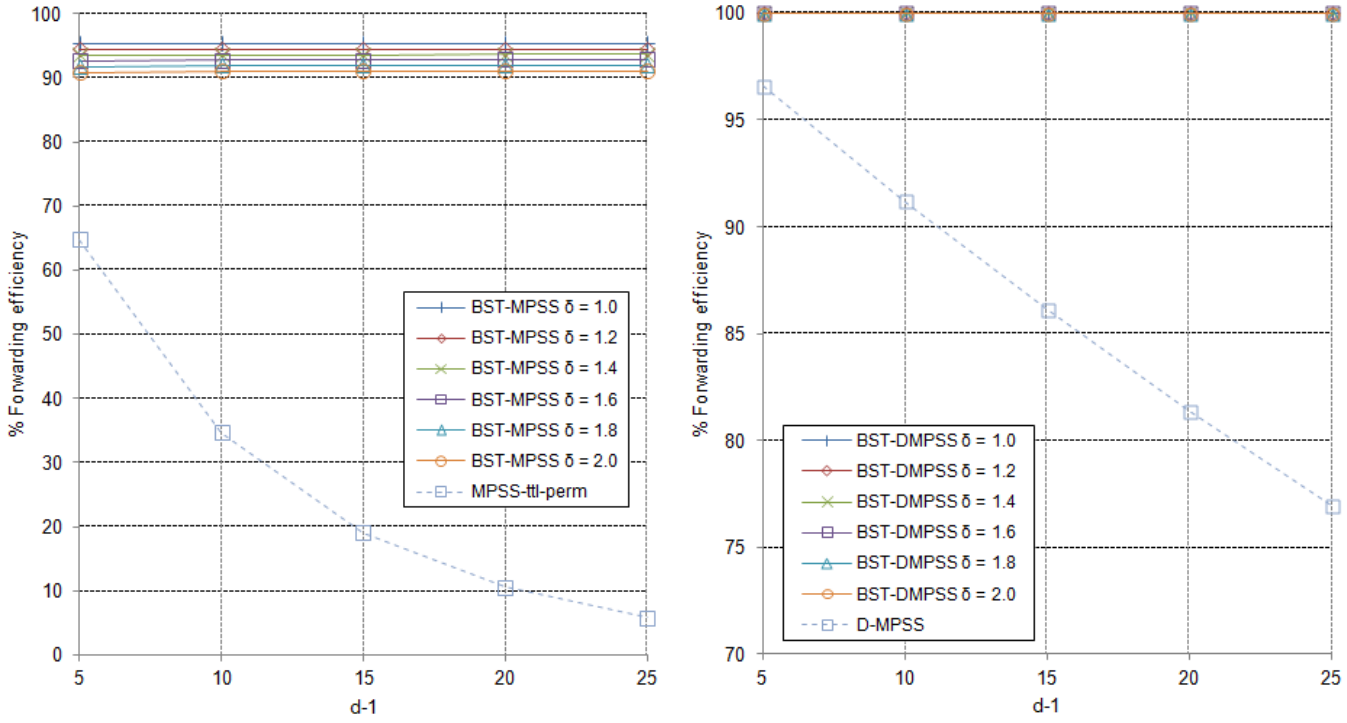
In order to evaluate the performance, the *forwarding efficiency* (fwe) [13] of any technique is

$$fwe = \frac{n}{n + oh} \quad (12)$$

Let us consider a multicast tree with $v = 2$ and $h = 6$, which makes $n = 126$ the number of edges of the tree. In Fig. 9a we provide the calculations of the forwarding efficiency for BST-MPSS, with different values of d and δ, and comparing it with the native MPSS. As expected, analytical results show clearly the great reduction of overhead due to the additional BST constraint, leading to an important improvement of the forwarding efficiency.

In the case of using D-MPSS as the inner BF technique, we have that the total overhead given by Eq. 13

$$totalOvh_{BST-DMPSS} = \sum_{i=1}^{h} \left(\prod_{j=1}^{i} fpr_j \cdot fanout(j)_{BST} \right) \quad (13)$$



(a) BST with MPSS, m=800, k=4

(b) BST with D-MPSS, M=800, k=6, mult=32

Fig. 9. Forwarding efficiency of BST for a multicast tree with $v = 2$, $h = 6$ over a regular network with a constant node degree of d and different values of δ (multiplier factor of v to determine the number of nodes N).

Let us consider the same multicast tree described before. Fig. 9b provides the analyticial results of the forwarding efficiency for BST with DMPSS, with different values of d and δ, and comparing it with D-MPSS. In general, hybrid BST aggregation makes the forwarding efficiency higher than 99%, outperforming D-MPSS and BST-MPSS methods.

2) Number of Forwarding Entries: It is not possible to calculate analytically the number of forwarding entries for hybrid techniques that involve the use of fit shared trees (FST-MPSS and FST-DMPSS), since the creation of STs is not a trivial task (the RP of an ST is actually the core node that gives the average shortest distance in number of hops from the RP to the rest of nodes). Also the state entries for the LSP from the source PE node to the RP node must be considered. But in the case of BST-MPSS and BST-DMPSS this formulation is possible. Let $|PA|$ be the total number of BSTs built (one per each PA aggregation node as RPs) and E the total number of unidirectional links (which gives the number of forwarding entries of the inner native BF technique). The total number of state entries over the network is

$$totalState_{BST} = |PA| \cdot (N - 2) + E \tag{14}$$

3) Header Overhead: The header overhead is calculated in a similar manner to the equation provided by [7] for the case of MPSS, with the only difference that now the 32 bits of the MPLS label (for the signaling of the ST or the BST) has to be added. Thus, with the same regular network and the same regular tree we have that

$$headerOvh_{hybrid} = (m + 32) \cdot n \tag{15}$$

For the hybrid aggregation - D-MPSS technique, the formulation presented in [7] for the D-MPSS case needs to be reformulated, obtaining:

$$headerOvh_{hybrid} = \sum_{i=2}^{h} \left(32 \cdot n_i + (m_i + s) \sum_{j=1}^{i-1} n_j \right) \tag{16}$$

IV. Simulations and Results

The simulations have been run over diverse reference networks such as NSFNET, Abilene, KPN (Europe), Tiscali (World) and COST-266. We only present the results obtained for NSFNET, because of space limitations and since the results obtained with the other networks were really similar and did not differ from the expected. All the network topologies were extended according to the provider aggregation model described in [12], thus resulting, in the case of NSFNET, 107 PE nodes 282 links. Simulations were run for testing all the hybrid techniques presented in this work under the following conditions. PE routers provide Internet and Layer 3 MPLS VPN services from these major locations. We generated 1,000 random sets of VPN samples with uniform distributions for each iteration. In all cases we used packets of 1,000 bytes. Simulations were run under Java 1.7.0_03 using the IDE

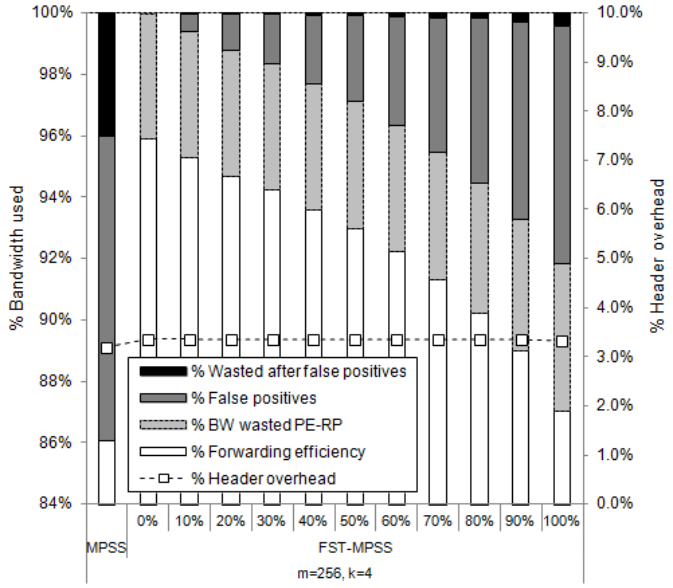


Fig. 10. Bandwidth used and header overhead of the MPSS and FST-MPSS approaches (NSFNET network, *m*=256, *k*=4).

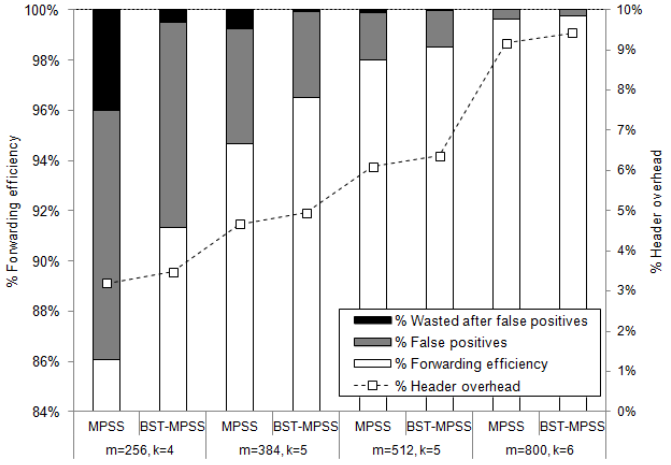


Fig. 11. Bandwidth used and header overhead of the BST-MPSS and MPSS with permutation techniques (NSFNET network).

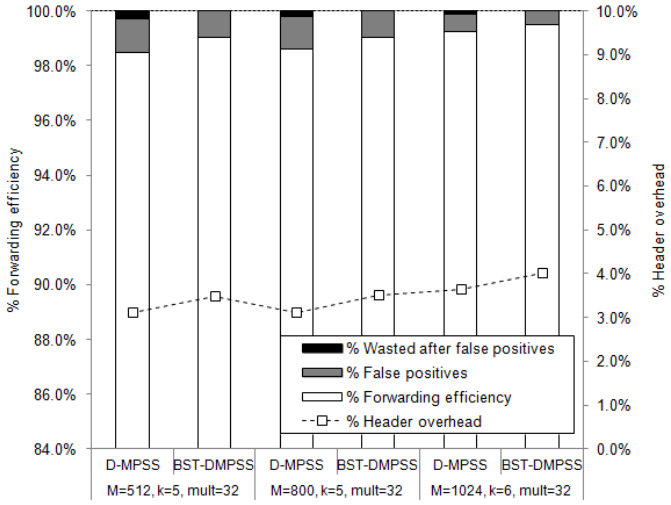


Fig. 12. Bandwidth used and header overhead of the BST-DMPSS and D-MPSS techniques (NSFNET).

Eclipse Juno, and were stopped when the target average X and confidence interval of 95%, $X \pm \Delta X$, held: $\Delta X/|X| \leq 5\%$ (for the sake of clarity, intervals are not represented in figures). For fair comparisons MPSS has been implemented with the bit permutation method [4] but not including the *time-to-live* (TTL) field in the header, since it is not necessary in an environment where shared trees are used (because any packet storm would be stopped as soon as it reaches one of the ST leaves, signaled by the MPLS labels). Since the FST method is not source-rooted, there is an amount of bandwidth wasted for sending packets from the PE sources to the rendesvouz point node of the ST, what is taken into account.

A. Bandwidth Used, Header Overhead and Forwarding Efficiency

In Fig. 10 we can see that, although the FST hybrid approach reduces (and for low levels of aggregation ratio (AR), it eliminates) the bandwidth wasted after false positives, it has the drawback of using a fair amount of bandwidth to deliver packets from PE sources to RP nodes of the FSTs. However, even for high values of AR, where only a few STs are built (therefore, less state is needed), these approach outperforms MPSS (with short filters of $m = 256$). Therefore, FST-MPSS represents a viable alternative when the global percentage of header overhead means an issue for the network performance and it is necessary to lower it, because in this cases it does not represent more than 5% of the traffic. As it was expected from the analytical formulations (Eq. 11), although fpr does not change in the shared tree scenario, the overhead that follows the false positives are minimized and almost withdrawn, thanks to the hybrid technique.

Regarding Broadcast Shared Trees with MPSS (BST-MPSS), Fig. 11 shows how the BST model contributes to save the bandwidth wasted for PE-RP transmissions, because BST roots are located at one hop of distance from PE nodes. The results gathered for BST with D-MPSS (BST-DMPSS) (Fig. 12) follow the same behavior as the previous one, with the only difference that, since the inner Bloom filter-based method (D-MPSS) performs better (better forwarding efficiency and lower header overhead), the global results are also improved thanks to the BSTs. It should be noted that from all of the above techniques tested, this one performs the best, regarding both forwarding efficiency and the load of header overhead. The same results were obtained in networks with different topologies and with different sizes. Thus, BST-DMPSS with $M = 1024$, $k = 6$, and $mult = 32$ could be used by a VPN Service Provider using this technique to support multicast communications with only less than 1% of bandwidth wasted and 4% of header overhead.

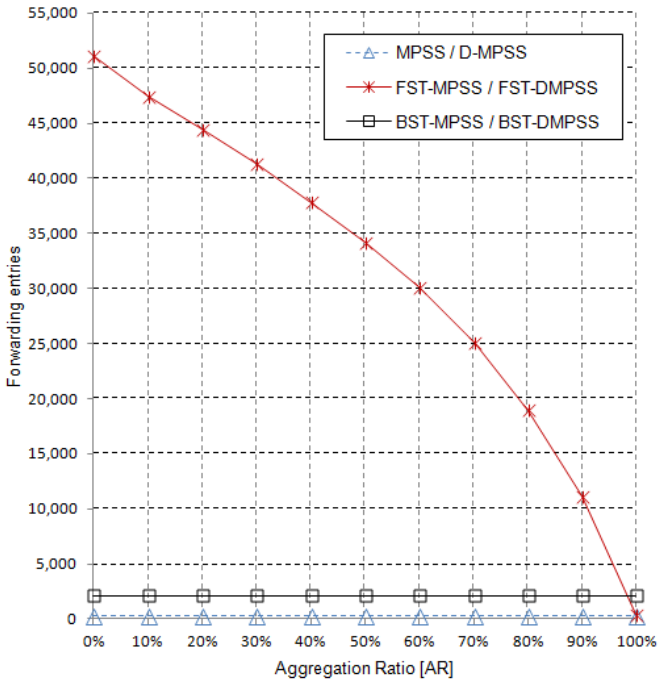


Fig. 13. Total number of state forwarding entries for native and hybrid techniques (NSFNET network).

B. Forwarding State Entries

The number of state forwarding entries at intermediate nodes to forward the whole set of VPNs samples through shared trees is summarized in Fig. 13. These figures also represent the state for the MPSS and D-MPSS cases, which are expected to be very low (each node has to maintain only one entry per outgoing link). As we can see, FSTs show acceptable results from $AR = 80\%$, which would make the bandwidth and forwarding efficiency metrics get worse. Also observe that the BST model does show a very low amount level of state too (in this simulation, below 20 entries per node in average), in coherence with the analytical results (Eq. 14). In this sense, the BST model provides a good trade-off between the forwarding efficiency, header overhead and state information, although the extra processing for the MPLS label.

V. Conclusions

In this paper we have proposed and studied several techniques for Hybrid Aggregation - Bloom filter-based forwarding techniques for multicast traffic in a VPN service provider network scenario. These techniques have been studied under two models for shared tree construction: fit shared trees (FST) and broadcast shared trees (BST), and with two BF forwarding mechanisms: MPSS and D-MPSS. The results show that the combination of BF and ST mean a relevant improvement in forwarding efficiency, removing most overhead traffic and cancelling loops. Regarding the shared tree models, FST presents the drawback of having to use an amount of links for sending packets from the provider edge nodes (PE nodes) to the corresponding roots of the shared trees, which initiate the multicast transmission, and the amount of forwarding entries is higher. Therefore, we advocate for the BST model of aggregation. In this approach, broadcast shared trees are built, locating their rendezvous points (RP) near the PE nodes, typically one hop before the PE node at an aggregation node, in order to save PE-RP bandwidth. The results showed that this hybrid aggregation method really contributes to improve the Bloom filter-based techniques and also limit the amount of the state forwarding entries to very low levels, as given by the few pre-configured broadcast shared trees, independently of the number of multicast groups or VPLS instances.

Acknowledgment

The authors would like to acknowledge the support of projects TIGRE5-CM (grant no. S2013/ICE-2919), TEXEO (grant no. TEC2016-80339-R) and Elastic Networks (TEC2015-71932-REDT) to the development of this work.

References

[1] I. Martinez-Yelmo, D. Larrabeiti, and I. Soto, "Multicast traffic aggregation in mpls-based vpn networks," *IEEE Communications Magazine*, vol. 45, no. 10, pp. 78–85, 2007.

[2] G. M. Fernandez, D. Larrabeiti, and J. A. de la Fuente, "On forwarding state control in vpn multicast based on mpls multipoint lsps," in *2012 IEEE 13th International Conference on High Performance Switching and Routing*, 5 2012, pp. 133–140.

[3] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," *Journal of Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2005.

[4] M. Sarela, C. E. Rothenberg, T. Aura, A. Zahemszky, P. Nikander, and J. Ott, "Forwarding anomalies in bloom filter-based multicast," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, Shangai, China, 2011, pp. 2399–2407.

[5] X. Tian and Y. Cheng, "Loop mitigation in bloom filter based multicast: A destination-oriented approach," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, Orlando, FL, USA, 2012, pp. 2131–2139.

[6] A. Zahemszky, P. Jokela, M. Sarela, S. Ruponen, J. Kempf, and P. Nikander, "Mpss: Multiprotocol stateless switching," in *Proceedings of the IEEE Conference on Computer Communications INFOCOM Workshops*, San Diego, CA, 2010, pp. 1–6.

[7] G. M. Fernandez and D. Larrabeiti, "Depth-wise multi-protocol stateless switching of multicast traffic," in *Proceedings of the IEEE Latin American Conference on Communications (LATINCOM)*, vol. (Submitted for acceptance), Cuenca, Ecuador, 2012.

[8] R. B. Martinez-Aguilar and G. M. Fernandez, "Implementation of stateless routing mechanisms for multicast traffic on netfpga card," in *Proceedings of the IEEE Colombian Conference on Communication and Computing (IEEE COLCOM 2015)*. IEEE, 2015, pp. 1–5.

[9] C. Rothenberg, P. Jokela, P. Nikander, M. Sarela, and J. Ylitalo, "Self-routing denial-of-service resistant capabilities using in-packet bloom filters," in *Proceeding of the European Conference on Computer Network Defense*, IEEE, Ed. IEEE, 2009, pp. 46–51.

[10] M. Sarela, C. E. Rothenberg, A. Zahemszky, P. Nikander, and J. Ott, "Bloomcasting: Security in bloom filter based multicast," *Lecture Notes in Computer Science (15th Nordic Conference on Secure IT Systems, Revised Selected Papers)*, vol. 7127, pp. 1–16, 2010.

[11] M. Antikainen, "On the security of in-packet bloom-filter forwarding (master thesis). aalto university. espoo, finland," Master's thesis, Aalto University, Espoo, Finland, June June 2011.

[12] J. Guichard, F. L. Faucheur, and J.-P. Vasseur, *Definitive MPLS Network Designs*. Cisco Press, 2005.

[13] P. Jokela, A. Zahemszky, C. E. Rothenberg, S. Arianfar, and P. Nikander, "Lipsin: Line speed publish/subscribe inter-networking," in *Proceedings of the ACM SIGCOMM Conference of the Special Interest Group on Data Communications*, ACM, Ed., Barcelona, Spain, 2009, pp. 195–206.