June 2021

MULTICAST VIRTUAL PRIVATE NETWORK PER-FLOW MONITORING FOR AN AGGREGATED TUNNEL IN A MULTIPROTOCOL LABEL SWITCHING CORE

Mankamana Mishra

Ashok Kumar

Sridhar Santhanam

Rajiv Asati

Nitin Kumar

MULTICAST VIRTUAL PRIVATE NETWORK PER-FLOW MONITORING FOR AN AGGREGATED TUNNEL IN A MULTIPROTOCOL LABEL SWITCHING CORE

AUTHORS:
Mankamana Mishra
Ashok Kumar
Sridhar Santhanam
Rajiv Asati
Nitin Kumar

ABSTRACT

Typically, when a service provider carries customer multicast traffic over a core network, it is often carried over a tunnel, which are often aggregated. There are many reasons for aggregated tunnel use, such as issues of scale and/or hardware limitations. While aggregated tunnels can be useful for carrying multicast traffic, it can be difficult to monitor network health when tunnels are aggregated. Techniques of this proposal provide for the ability to monitor network health by supporting per-flow counters for aggregated tunnels for both Internet Protocol (IP) version 4 and version 6 (IPv4/IPv6) traffic in a manner that is scalable and can be provided on-demand.

DETAILED DESCRIPTION

There are many reasons why a service provider may carry multicast traffic over aggregated tunnels. For example, in some cases customer multicast traffic could be very high (e.g., in the order of hundreds of thousands of flows) and a service provider may not want to create per-customer flow states in the core of the network. For example, if there are 'n' customers and 'm' flows per customer, the core would end up having 'n x m' states.

In another example, hardware limitations may be a reason for using aggregated tunnels. If per-flow tunnels are created, the per-customer number of flows would be multiplied by the total number of customers, which could also be on the order of hundreds of thousands of tunnels. Many platforms cannot support such a scale. One reason for having an aggregated tunnel in the core network is that there is no virtual routing and forwarding (VRF) context. However, there may be a need in some instances to differentiate flows for which a tunnel is creates at the head-end and tail-end for a flow.

Thus, service providers have a choice to make between optimality versus scalability and a majority of service providers opt for a scalable solution that involves a manageable scale rather than an optimal solution.

A Data Multicast Distribution Tree (MDT) provides details for how a multicast tree is formed. Consider an example network topology, as shown below in Figure 1.

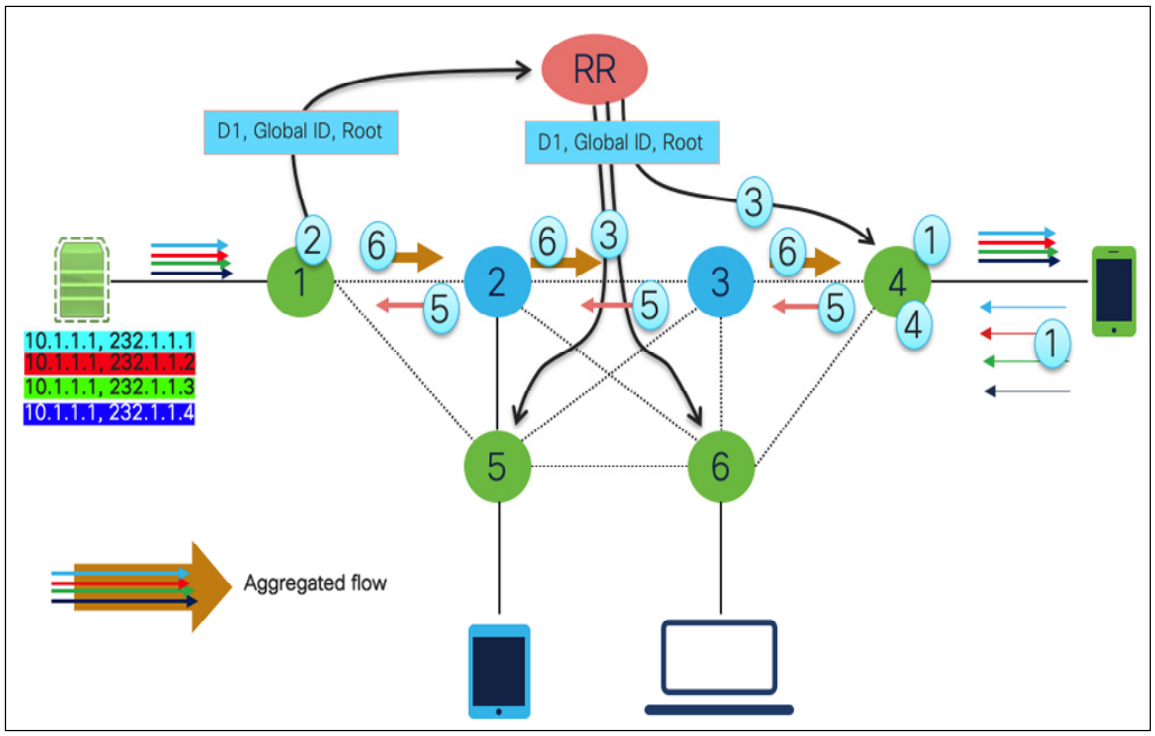

Figure 1: Example Network Topology

For the example network topology of Figure 1, consider that a provider edge (PE) router 1 is a head-end router and routers 5, 6, and 4 are tail-end routers. Further consider that there are four flows for the present example that are coming from a source behind PE 1 and that the source sending traffic and a receiver sending a join can occur in any order. Further, there are many types of trees that can be built. For the present example, consider that there is a threshold configured to move from one type of tree to another type of tree. However, the overall concepts of the proposal described herein remain the same. Additionally, various discussions herein consider a Multiprotocol Label Switching (MPLS)-based (e.g., Label Distribution Protocol (LDP) or Segment Routing (SR) data plane).

When the control plane setup is being performed, consider the following high-level steps:

1. Receiver sends IGMP join for all 4 flows.
2. Egress PE (2) uses Border Gateway Protocol (BGP)-based overlay signaling to notify the Ingress router about an interest. Ingress PE (1) determines the interest and decides that all of these flows are to aggregated to a tree (e.g., based on local configuration). It generates key which would be used as the Forward Equivalence Class (FEC) from the Egress node to setup the underlay tree.
3. The Ingress PE sends Data MDT information to all BGP speakers that carries the root IP address along with a generated ID that is associated with a customer (S,G). In this case, all four flows would have exact same (root, Data MDT number and global ID), which provides an indication to the Egress node that it needs to setup only 1 underlay tree for all 4 flows.
4. The Egress PE receives the allocation from the Ingress PE generates an MLDP FEC towards the root.
5. A hop-by-hop join towards the root is sent. If the same FEC is already present in system locally then only the outgoing port would be added.
6. Traffic flow starts along the path of tree. Details for the aggregated flows for this example are shown below in Figure 2. When considering routers 2 and 3, these routers play the role routers and completely operate on the Transport label.

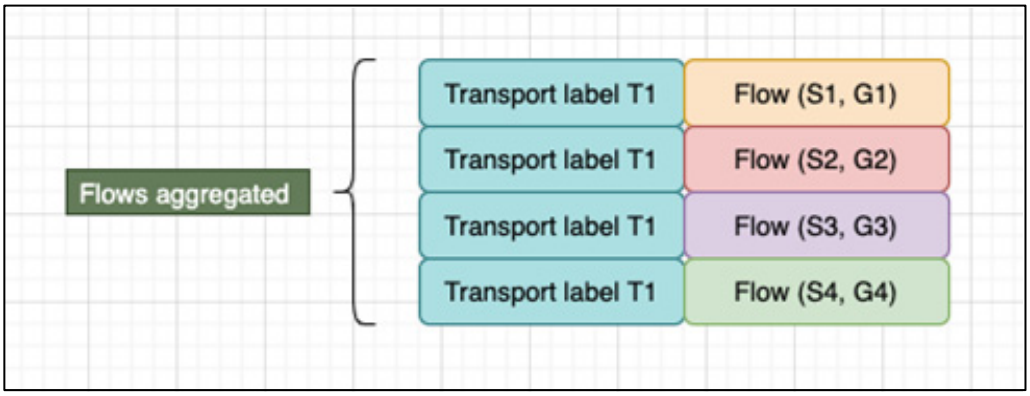

Figure 2: Example Flow Details

There are many different third-party tools that service providers can use to monitor overall network health, however, such monitoring typically involves per-flow statistics (stats) to be exported from the network exported to a given tool. However, currently, a core router for the example shown above does not have the ability to provide per-flow stats as it is routing based on the aggregated tunnel and best counters that can be provided would be per-transport label counts. To provide per-flow counts, each node would need to go and perform a lookup inside packets to identify flow information. However, this still will be insufficient, as the same flow can be coming in different VRFs and there needs to be some differentiator that can differentiate two different VRFs carrying the same flow.

Various potential solutions to this issue may include using deep packet inspection to perform per-flow lookups or using span port to provide off-box accounting. However, deep packet inspection will likely not scale well as each core node will need to forward the traffic and also perform an IP lookup to count packets per flow. Since the same flow can be present in different VRFs and there is no VRF knowledge in core, this is another challenge. Still, one of the biggest challenges would be how to communicate in whole network which tunnel to monitor. If monitoring is turned on for all flows, the platform would run into scale issues. Regarding span port to perform off-box accounting, one port can be marked as a span port and traffic can be exported out. However, this is also not a scalable solution for deployments having hundreds of thousands of flows and devices that continue to grow; it is not feasible to have one extra port and device just to account and monitor traffic.

In light of these challenges and issues, techniques of this proposal provide for the ability to monitor network health by supporting per-flow counters for aggregated tunnels for both IPv4/IPv6 traffic in a manner that is scalable and can be provided on-demand. Broadly, techniques of this proposal provide for:

1. Identifying flows to monitor;
2. Using in-band signaling to notify each hop in the network to start per-flow accounting;
3. Classifying traffic from an Ingress such that per-flow monitoring can be performed in the whole network; and
4. Providing per-flow monitoring and reporting by each node in the network.

Consider Figure 3, below, through which various details of the techniques of this proposal are described in order to facilitate network health monitoring and reporting.

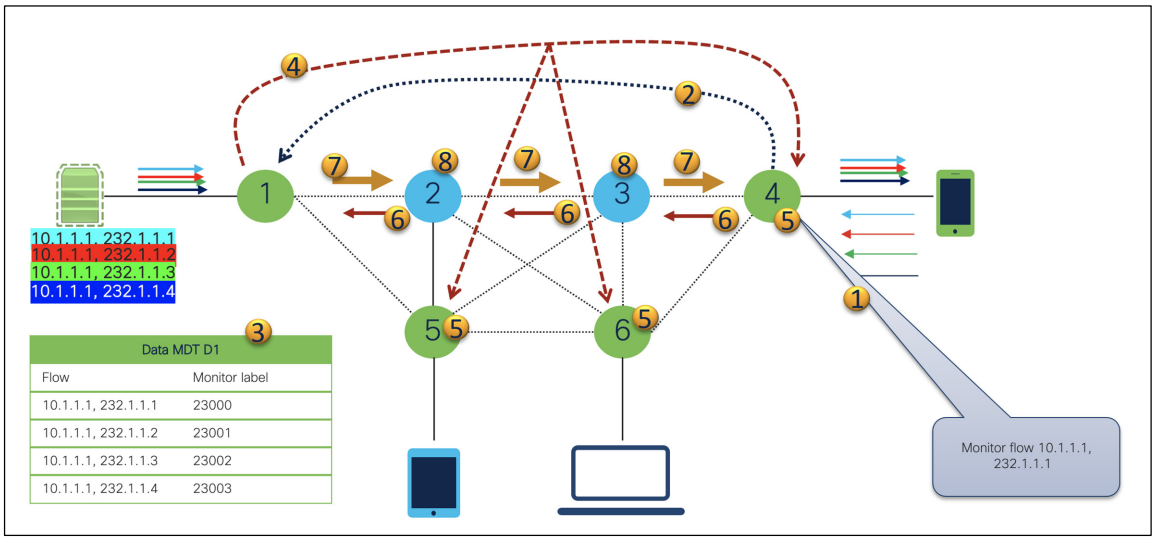

Figure 3: Example Framework to Facilitate Network Health Monitoring and Reporting

The example of Figure 3 illustrates that the process to provide network health monitoring and reporting is driven from the Egress node, however, there is nothing that would prohibit the process to be started from the Ingress node, in which case Step-3 and onwards, as shown below, would be applicable. Consider various example steps that can be utilized to facilitate the monitoring and reporting of this proposal, as follows:

1. User identifies flow to be monitored;
2. An overlay join signals notification about the flow to be monitored, which carries an attribute such that it is not suppressed by a route reflector (RR) and is notified to the first hop router;
3. The first hop router now identifies the MDT tunnel to which the flow belongs and allocates label for all flows associated with this MDT tunnel;
4. A Selective P-Multicast Service Interface (S-PMSI) carries information that these flow will be coming with an extra label that is being used for monitoring;
5. Each Egress node can decide whether to monitor their part of network, even if there is no monitoring occurring (they do not want to do rest other process

which is Step-6 onwards). Each Egress node needs to program hardware such that the 2nd label is popped before any other lookup is performed;

6. A Hop-by-Hop join carries a flag that means that monitoring is expected;
7. Once an underlay signal reaches the Ingress, the Ingress starts sending traffic with two labels, which are a transport and a monitoring label;
8. Each node now takes counter for the per-monitoring label.

At Step-8, while performing a look-up in order to increment a counter, it must be keyed by (Transport label + Inner label). This ensures that explicit signaling is not needed to make the unique inner (monitoring) label across different roots for different VRFs.

For example, consider a case where router 1 is sourcing 4 flows (as shown in Figure 4A) and for VRF RED and router 5 is sourcing the exact same pair of (S,G) on VRF BLUE (as shown in Figure 4B). Since router 1 and router 5 do not talk to each other about the monitoring label assignment, they can potentially assign same label.

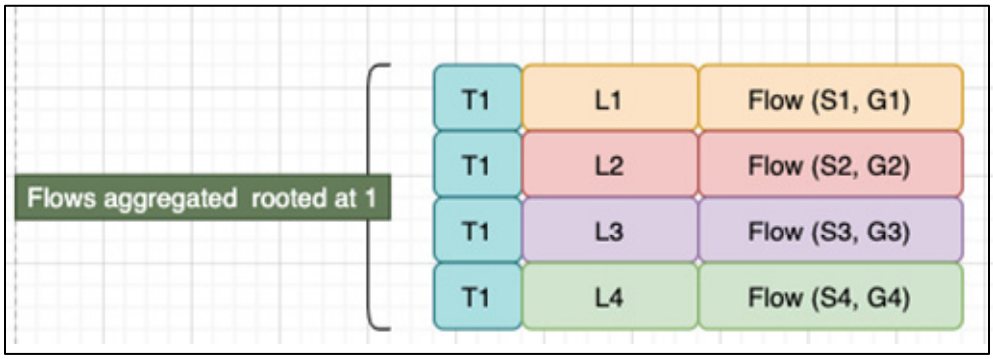

Figure 4A: Aggregated Flows at Router 1

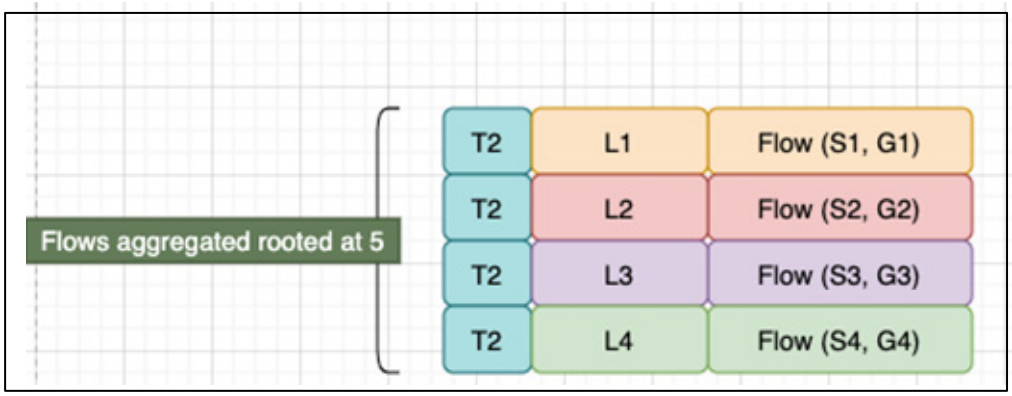

Figure 4A: Aggregated Flows at Router 2

For the above example, when router 2 tries to take a count, it is going to take in a pair of Transport label and Internal label, which provides a unique flow counter that can be exported to an external server. Without the (Transport label + internal label) pair, it would be hard to identify which VRF flow to which a count belongs. Since a per-root, per-tunnel different transport label would be used, flexibility is provided to identify unique flow monitoring.

Consider another example use case, as shown below via Figure 5, involving partial flow monitoring.

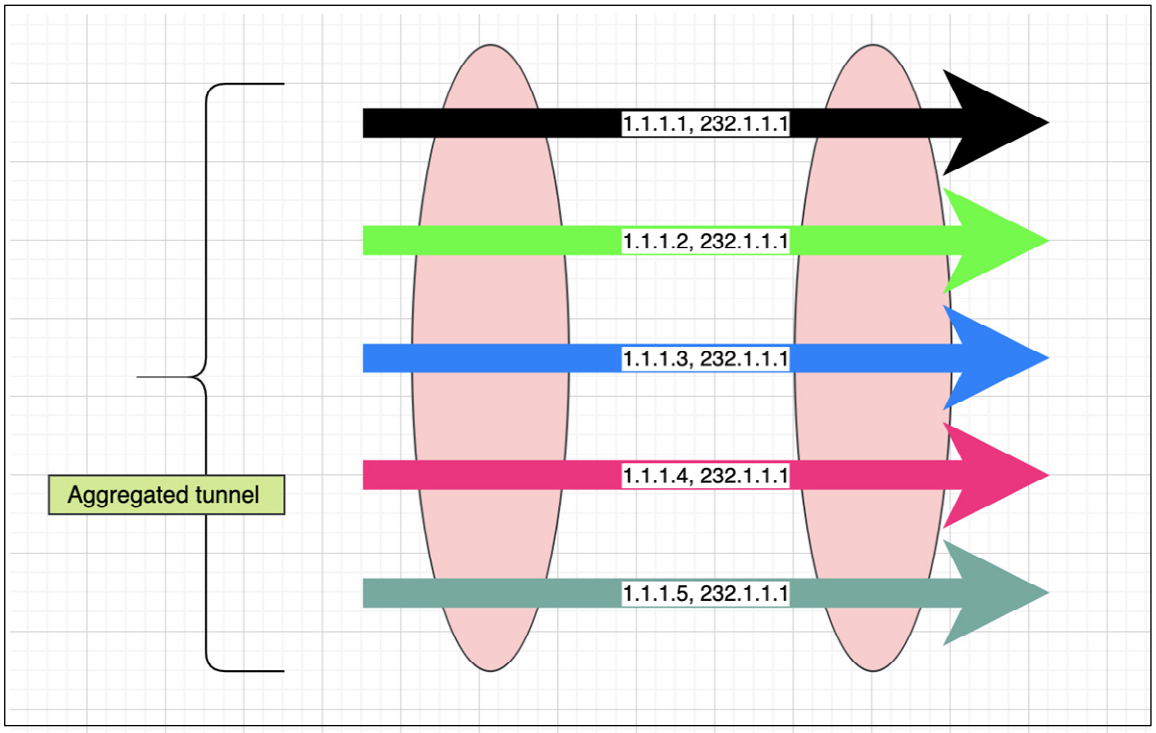

Figure 5: Partial Flow Monitoring Example

For the example illustrated in Figure 5, consider that 5 flows are being aggregated into single tunnel, but due to nature of a given application a provider may only be interested in monitoring a subset of flows. Thus, in this example use-case an implementation can assign a unique label to only those flows that need to be monitored and a shared label can be used for the other. This would avoid having per-flow state for monitoring in core of the network that may not be needed.

Considering a hybrid topology and potential innovation impacts and a theoretical assumption that if there is some node in an MVPN network that does not support the

techniques of this proposal that there may be an issue with implementing the proposal. However, with respect to signaling, a new flag can be introduced in an Administrative Distance (AD) route where capability is exchanged. If there is one node that does not support the techniques of this proposal, it would not be turned on. This would help even in a network upgrade scenario where system is being upgraded in phases. If a service provider is planning to achieve per-flow monitoring in whole network, any new mechanism would involve a software upgrade across network. Thus, practically, this could be new that feature could be introduced in the network.

Accordingly, this proposal provides per-flow monitoring techniques that can be applied for all MLDP-based MVPN profiles that are address family agnostic. Further, monitoring can be turned-on or turned-off on-demand. Further, no limitations are involved for cases in which the same flow is present in multiple VRF instances. This framework can further support any hardware extensions to calculate jitter, packet delay, etc. for hardware that is capable of making such measurements.