SPRING Working Group
Internet-Draft
Intended status: Informational
Expires: June 20, 2015

P. Lapukhov
E. Aries
G. Nagarajan
Facebook

December 17, 2014

Use-Cases for Segment Routing in Large-Scale Data Centers
draft-lapukhov-segment-routing-large-dc-00

Abstract

This document discusses ways in which segment routing (aka source routing) paradigm could be leveraged inside the data-center to improve application performance and network reliability. Specifically, it focuses on exposing path visibility to the host's networking stack and leveraging this to address a few well-known performance and reliability problems in data-center networks.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft will expire on June 20, 2015.

Copyright Notice

Table of Contents

1. Introduction

The source routing principles decscribed in [I-D.filsfils-spring-segment-routing] allow applications to have fine-grained control over their traffic flow in the network. Traditionally, path selection was performed solely by the network devices, with the end-host only responsible for picking ingress points to the data-center network (selecting first-hop gateways). If the network devices support source-routed instructions of some kind (e.g. encoded in MPLS labels), the end-hosts would benefit from knowing about all possible paths to reach a network destination. This enables the networking stack to route over different paths to the same prefix, based on relative performance of each path, or perform more complex load-distribution, e.g. not necessarily of equal-cost.

Source-routing has known trade-offs, such as requiring the hosts to maintain more information about the network and keeping this information up-to-date. These trade-offs need to be considered and addressed when building the actual production system based on source-routed principles. Design of such systems is outside of the scope of this document, which concerns mostly with the use-cases that are possible within source-routed paradigm.

2. Large-scale data center network design summary

This section provides a brief summary of the informational document [I-D.ietf-rtgwg-bgp-routing-large-dc] that outlines a practical network design suitable for data-centers of various scales.

- Data-center networks have highly symmetric topologies with multiple parallel paths between two server attachment points. The well-known Clos topology is most popular among the operators. In a Clos topology, the number of parallel paths between two elements is determined by the "width" of the middle stage. See Figure 1 below for an illustration of the concept.

- Large-scale data-centers commonly use a routing protocol, such as BGPv4 [RFC4271] to provide endpoint connectivity. Recovery after a network failure is therefore driven either by local knowledge of directly available backup paths or by distributed signaling between the network devices.

- Within data-center networks, traffic is load-shared using the Equal Cost Multipath (ECMP) mechanism. With ECMP, every network device implements a pseudo-random decision, mapping packets to one of the parallel paths by means of a hash function calculated over certain parts of the packet, typically some packet header fields.

The following is a schematic of a five-stage Clos topology, with four devices in the middle stage. Notice that number of paths between "DEV A" and "DEV L" equals to four: the paths have to cross all of Tier-1 devices. At the same time, the number of paths between "DEV A" and "DEV B" equals two, and the paths only cross Tier-2 devices. Other topologies are possible, but for simplicity we'll only look into the topologies that have a single path from Tier-1 to Tier-3. The rest could be treated similarly, with a few modifications to the logic.

```
                                Tier-1
                               +-----+
                               | DEV |
                            +--|  E  |--+
                            |  +-----+  |
                    Tier-2  |           |   Tier-2
                   +-----+  |  +-----+  |  +-----+
     +-------------| DEV |--+--| DEV |--+--| DEV |-------------+
     |       +-----|  C  |--+  |  F  |  +--|  I  |-----+       |
     |       |     +-----+     +-----+     +-----+     |       |
     |       |                                         |       |
     |       |     +-----+     +-----+     +-----+     |       |
     | +-----+-----| DEV |--+  | DEV |  +--| DEV |-----+-----+ |
     | |     | +---|  D  |--+--|  G  |--+--|  K  |---+ |     | |
     | |     | |   +-----+  |  +-----+  |  +-----+   | |     | |
     | |     | |            |           |            | |     | |
   +-----+ +-----+          |  +-----+  |          +-----+ +-----+
   | DEV | | DEV |          +--| DEV |--+          | DEV | | DEV |
   |  A  | |  B  | Tier-3      |  H  |      Tier-3 |  L  | |  M  |
   +-----+ +-----+             +-----+             +-----+ +-----+
     | |     | |                                     | |     | |
     O O     O O                                     O O     O O
       Servers                                         Servers
```

Figure 1: Five-Stage Clos topology

3. Some open problems in large data-center networks

The data-center network design summarized above provides means for moving traffic between hosts with reasonable efficiency. There are few open performance and reliability problems that arise in such design:

- ECMP routing is most commonly realized per-flow. This means that large, long-lived "elephant" flows may affect performance of smaller, short-lived flows and reduce efficiency of per-flow load-sharing. In other words, per-flow ECMP that does not perform efficiently when flow life-time distribution is heavy-tailed. Furthermore, due to hash-function inefficiencies it is possible to have frequent flow collisions, where more flows get placed on one path over the others.

- Shortest-path routing with ECMP implements oblivious routing model, which is not aware of the network imbalances. If the network symmetry is broken, for example due to link failures, utilization hotspots may appear. For example, if a link fails between Tier-1 and Tier-2 devices (e.g. "DEV E" and "DEV I"),

Tier-3 devices "DEV A" and "DEV B" will not be aware of that, since there are other paths available from perspective of "DEV C". They will continue sending traffic as if the failure didn't exist and may cause a traffic hotspot.

- Absence of path visiblity leaves transport protocols, such as TCP, with a "blackbox" view of the network. Some TCP metrics, such as SRTT, MSS, CWND and few others could be inferred and cached based on past history, but those apply to destinations, regardless of the path that has been chosen to get there. Thus, for instance, TCP is not capable of remembering "bad" paths, such as those that exhibited poor performance in the past. This means that every new connection will be established obliviously (memory-less) with regards to the paths chosen before, or chosen by other nodes.

- Isolating faults in the network with multiple parallel paths and ECMP-based routing is non-trivial due to lack of determinism. Specifically, the connections from host A to host B may take a different path every time a new connection is formed, thus making consistent reproduction of a failure much more difficult. This complexity scales linearly with the number of parallel paths in the network, and stems from the random nature of path selection by the network devices.

Further in this document, we are going to demonstrate how these problems could be addressed within the framework of a source-routing model.

4. Augmenting the network with segment routing

Imagine a data-center network equipped with some kind of segment-routing signaling, e.g. using [I-D.keyupate-idr-bgp-prefix-sid]. The end-hosts in such network may now specify a path for a packet, or a flow, by attaching a segment instruction (e.g. MPLS label stack) to the packet. For instance, when using MPLS data-plane, a label corresponding to the shortest route toward one of the Tier-1 devices could be attached to a packet. The packet would therefore be forced to go across the specific Tier-1 devices, which would pre-determine its end-to-end path inside the data-center given the properties of Clos topology. Note that in this case, the segment-routing directive will be stripped once the packet reaches the Tier-1 device, and remaining forwarding will be done using regular IP lookups.

As a result, the hosts become aware of the path that their packets would take. The hosts no longer have to rely on oblivious ECMP hashing in the network to select a random path, but may choose between "deterministic" or "random" routing, where randomness is

controlled by the hosts via "random" choice of the segment-routing directives.

Note that under this proposal, the segment routing signaling and the corresponding data-plane component "augment" existing IP forwarding mechanisms, but do not necessarily fully replace it. This allows for gradual deployment and testing of the new functionality with a simple rollback strategy. In addition, this allows to keep existing operational procedures, such as those involving shifting traffic on/ off the boxes/links by involving routing protocol manipulations.

5. Communicating path information to the hosts

There are two general methods for communicating path information to the end-hosts: "proactive" and "reactive", aka "push" and "pull" models. There are multiple ways to implement either of these methods. Here, we note that one way could be using a centralized agent: the agent either tells the hosts of the prefix-to-path mappings beforehands and updates them as needed (network event driven push), or responds to the hosts making request for a path to specific destination (host event driven pull). It is also possible to use a hybrid model, i.e. pushing some state in response to particular network events, while pulling the other state on demand from host.

We note, that when disseminating network-related data to the end-hosts a tradeoff is made to balance the amount of information vs the level of visibility in the network state. This applies both to push and pull models. In one corner case (complete pull) the host would request path information on each flow, and keep no local state at all. In the other corner case, information for every prefix in the network along with available paths is pushed and continuously updated on all hosts.

6. Addressing the open problems

This section demonstrates how the problems describe above could be solved using the segment routing concept. It is worth noting that segment routing signaling and dataplane are only parts of the solution. Additional enhancements, e.g. such as centralized controller mentioned before, and host networking stack support are required to implement the proposed solutions.

6.1. Per-packet and flowlet switching

With the ability to choose paths on the host, one may go from per-flow load-sharing in the network to per-packet or per-flowlet (see [KANDULA04] for information on flowlets). The host may select different segment routing instructions either per packet, or per-

flowlet, and route them over different paths. This allows for solving the "elephant flow" problem in the data-center and avoiding link imbalances.

Note that traditional ECMP routing could be easily simulated with on-host path selection, using method proposed in VL2 (see [GREENBERG09]). The hosts would randomly pick up a Tier-2 or Tier-1 device to "bounce" packet off of, depending on whether the destination is under the same Tier-2 switches, or has to be reached across Tier-1. The host would use hash-function that operates on per-flow invariants, to simulate per-flow load-sharing in the network.

6.2. Performance-aware routing

Knowing the path associated with flows/packets, the end host may deduce certain characteristics of the path on its own, and additionally use the information supplied with path information pushed from the controller or received via pull request. The host may further share its path observations with the centralized agent, so that the latter may keep up-to-date network health map and assist other hosts with this information.

For example, in local case, if a TCP flow is pinned to a known path, the hosts may collect information on packet loss, deduced from TCP retransmissions and other signals (e.g. RTT inreases). The host may additionally publish this information to a centralized agent, e.g. after a flow completes, or by periodically sampling it. Next, using both local and/or global performance data, the host may pick up the best path for the new flow, or update an existing path (e.g. when informed of congestion on an existing path).

One particularly interesting instance of performance-aware routing is dynamic fault-avoidance. If some links or devices in the network start discarding packets due to a fault, the end-hosts would detect the path(s) being affected and steer their flows away from the problem spot. Similar logic applies to failure cases where packets get completely black-holed, e.g. when a link goes down.

6.3. Non-oblivious routing

By leveraging source routing, one avoids issues associated with oblivious ECMP hashing. For example, if in the topology depicted on Figure 1 a link between "DEV E" and "DEV I" fails, the hosts may exclude the segment corresponding to "DEV E" from the prefix matching the servers under Tier-2 devices "DEV I" and "DEV K". In the push path discovery model, the affected path mappings may be explicitly pushed to all the servers for the duration of the failure. The new

mapping would instruct them to avoid the particular Tier-1 switch until the link has recovered. Alternatively, in pull path discovery model, the centralized agent may start steering new flows immediately after it discovers the issue. Until then, the existing flows may recover using local detection of the path issues, as described in Section 6.2.

6.4. Deterministic network probing

Active probing is a well-known technique for monitoring network elements health, constituting of sending continuous packet streams simulating network traffic to the hosts in the data-center. Segment routing makes possible to prescribe the exact paths that each probe or series of probes would be taking toward their destination. This allows for fast correlation and detection of failed paths, by processing information from multiple actively probing agents. This complements the data collected from the hosts routing stacks as described in Section 6.2 section.

For example, imagine a probe agent sending packets to all machines in the data-center. For every host, it may send packets over each of the possible paths, knowing exactly which links and devices these packets will be crossing. Correlating results for multiple destinations with the topological data, it may automatically isolate possible problem to a link or device in the network.

7. Routing traffic outside of data-center

This document purposely does not discuss the multitude of use cases outside of data center center. However, it is important to note that source routing concept could be used to construct uniform control and data-plane for both data-center and Wide Area Network (WAN). Source routing instruction could be used in the end hosts to direct traffic outside of the datacenter, provided that all elements in the path support the corresponding data-plane instructions. For example, the model proposed in [I-D.filsfils-spring-segment-routing-central-epe] could be implemented under the same network stack modifications that are needed for the data-center use cases. In addition to the edge case, some sort the inter-DC traffic engineering could be realized by programming the end hosts. For illustration, an aggregate prefix for DC2 could be installed in all machines in DC1, enlisting all or some of the available paths (possibly with loose semantic) along with their performance characteristics. The exact algorithm for packet, flowlet or flow mapping to these paths is specific to a particular implementation.

Furthermore, visibility in the WAN paths allows the hosts to make more intelligent decisions and realize performance routing or fault avoidance approaches proposed for the data-center network above.

8. Conclusion

This document summarizes some use cases that segment/source routing model may have in a large-scale data-center. All of these are equally applicable to data-centers regardless of their scale, as long as they support the routing design implementing segment routing signaling.

9. IANA Considerations

TBD

10. Manageability Considerations

TBD

11. Security Considerations

TBD

12. Acknowledgements

TBD

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

13.2. Informative References

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[I-D.filsfils-spring-segment-routing]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.

[I-D.ietf-rtgwg-bgp-routing-large-dc]
Lapukhov, P., Premji, A., and J. Mitchell, "Use of BGP for routing in large-scale data centers", draft-ietf-rtgwg-bgp-routing-large-dc-00 (work in progress), August 2014.

[I-D.keyupate-idr-bgp-prefix-sid]
Patel, K., Ray, S., Previdi, S., and C. Filsfils, "Segment Routing Prefix SID extensions for BGP", draft-keyupate-idr-bgp-prefix-sid-00 (work in progress), October 2014.

[I-D.filsfils-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Patel, K., Aries, E., shaw@fb.com, s., Ginsburg, D., and D. Afanasiev, "Segment Routing Centralized Egress Peer Engineering", draft-filsfils-spring-segment-routing-central-epe-02 (work in progress), July 2014.

[KANDULA04]
Sinha, S., Kandula, S., and D. Katabi, "Harnessing TCP's Burstiness with Flowlet Switching", 2004.

[GREENBERG09]
Greenberg, A., Hamilton, J., Jain, N., Kadula, S., Kim, C., Lahiri, P., Maltz, D., Patel, P., and S. Sengupta, "VL2: A Scalable and Flexible Data Center Network", 2009.

Authors' Addresses

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Ebben Aries
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: exa@fb.com

Gaya Nagarajan
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: gaya@fb.com