Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 22, 2018

C. Filsfils, Ed.
S. Previdi, Ed.
Cisco Systems, Inc.
B. Decraene
Orange
R. Shakir
Google

December 19, 2017

Resiliency use cases in SPRING networks
draft-ietf-spring-resiliency-use-cases-12

Abstract

This document identifies and describes the requirements for a set of use cases related to network resiliency on Segment Routing (SPRING) networks.

Requirements Language

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT" and "MAY" in this document are used to define requirements for protocol and architecture design.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 22, 2018.

Copyright Notice

Table of Contents

1. Introduction

This document reviews various use cases for the protection of services in a SPRING network. The terminology used hereafter is in line with [RFC5286] and [RFC5714].

The resiliency use cases described in this document can be applied not only to traffic that is forwarded according to the SPRING architecture but also to traffic that originally is forwarded using other paradigms such as LDP signalling or pure IP traffic (IP routed traffic).

Three key alternatives are described: path protection, local protection without operator management and local protection with operator management.

Path protection lets the ingress node be in charge of the failure recovery, as discussed in Section 2.

The rest of the document focuses on approaches where protection is performed by the node adjacent to the failed component, commonly referred to as local protection techniques or Fast Reroute techniques ([RFC5286], [RFC5714]).

In Section 3 we discuss two different approaches providing unmanaged local protection, namely link/node bypass protection and shortest path based protection.

Section 4 illustrates a case allowing the operator to manage the local protection behavior in order to accommodate specific policies.

In Section 5 we discuss the opportunity for the SPRING architecture to provide loop-avoidance mechanisms, such that transient forwarding state inconsistencies during routing convergence do not lead into traffic loss.

The purpose of this document is to illustrate the different use cases and explain how an operator could combine them in the same network (see Section 6). Solutions are not defined in this document.

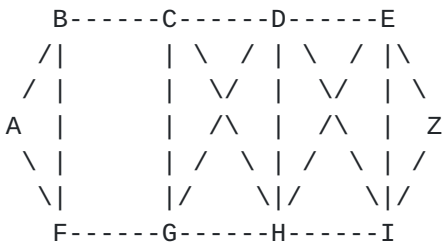

Figure 1: Reference topology

We use Figure 1 as a reference topology throughout the document. Following link metrics are applied:

Link metrics are bidirectional. In other words, the same metric value is configured at both side of each link.

Links from/to A and Z are configured with a metric of 100.

CH, GD, DI and HE links are configured with a metric of 6.

All other links are configured with a metric of 5.

2. Path Protection

As a reminder, one of the major network operator requirements is path disjointness capability. Network operators have deployed infrastructures with topologies that allow paths to be computed in a complete disjoint fashion where two paths wouldn't share any component (link or router) hence allowing an optimal protection strategy.

A first protection strategy consists of excluding any local repair but instead uses end-to-end path protection where each SPRING path is protected by a second disjoint SPRING path. In this case, the local protection is not used along the path.

For example, a Pseudo Wire (PW) from A to Z can be "path protected" in the direction A to Z in the following manner: the operator configures two SPRING paths T1 (primary) and T2 (backup) from A to Z.

The two paths may be used:

- concurrently, where the ingress router sends the same traffic over the primary and secondary path. This is usually known as 1+1 protection.

- concurrently, where the ingress router splits the traffic over the primary and secondary path. This is usually known as equal cost multi path (ECMP) or unequal cost multi path (UCMP).

- as a primary and backup path, where the secondary path is used only when the primary failed. This is usually known as 1:1 protection.

T1 is established over path {AB, BC, CD, DE, EZ} as the primary path and T2 is established over path {AF, FG, GH, HI, IZ} as the backup path. The two paths MUST be disjoint in their links, nodes and shared risk link groups (SRLGs) to satisfy the requirement of disjointness.

In the case of primary/backup paths, when the primary path T1 is up, the packets of the PW are sent on T1. When T1 fails, the packets of the PW are sent on backup path T2. When T1 comes back up, the operator either allows for an automated reversion of the traffic onto T1 or selects an operator-driven reversion. Typically, the switchover from path T1 to path T2 is done in a fast reroute fashion (e.g.: sub-50 milliseconds range) but depending on the service that needs to be delivered, other restoration times may be used.

It is essential that any path, primary or backup, benefit from an end-to-end liveness monitoring/verification. The method and mechanisms that provide such liveness check are outside the scope of this document. An example is given by [RFC5880].

There are multiple options for liveness check, e.g., path liveness where the path is monitored at the network level (either by the head-end node or by a network controller/monitoring system). Another possible approach consists of a service-based path monitored by the service instance (verifying reachability of the endpoint). All these options are given here as examples. While this document does express the requirement for a liveness mechanism, it does not mandate, nor define, any specific one.

From a SPRING viewpoint, we would like to highlight the following requirements:

- SPRING architecture MUST provide a way to compute paths that are not protected by local repair techniques (as illustrated in the example of paths T1 and T2).

- SPRING architecture MUST provide a way to instantiate pairs of disjoint paths on a topology based on a protection strategy (link, node or SRLG protection) and allow the validation or re-computation of these paths upon network events.

- The SPRING architecture MUST provide end-to-end liveness check of SPRING based paths.

3. Management-free Local Protection

This section describes two alternatives providing local protection without requiring operator management, namely bypass protection and shortest-path based protection.

For example, a traffic from A to Z, transported over the shortest paths provided by the SPRING architecture, benefits from management-free local protection by having each node along the path automatically pre-compute and pre-install a backup path for the destination Z. Upon local detection of the failure, the traffic is repaired over the backup path in sub-50 milliseconds. When the primary path comes back up, the operator either allows for an automated reversion of the traffic onto it or selects an operator-driven reversion.

The backup path computation SHOULD support the following requirements:

- o 100% link, node, and SRLG protection in any topology.
- o Automated computation by the IGP.
- o Selection of the backup path such as to minimize the chance for transient congestion and/or delay during the protection period, as reflected by the IGP metric configuration in the network.

3.1. Management-free Bypass Protection

One way to provide local repair is to enforce a fail-over along the shortest path around the failed component.

In case of link protection, the point of local repair will create a repair path avoiding the protected link and merging back to primary path at the nexthop.

In case of node protection, the repair path will avoid the protected node and merge back to primary path at the next-nexthop.

In case of SRLG protection, the repair path will avoid members of the same group and merge back to primary path just after.

In our example, C protects destination Z against a failure of CD link by enforcing the traffic over the bypass {CH, HD}. The resulting end-to-end path between A and Z, upon recovery against the failure of CD, is depicted in Figure 2.

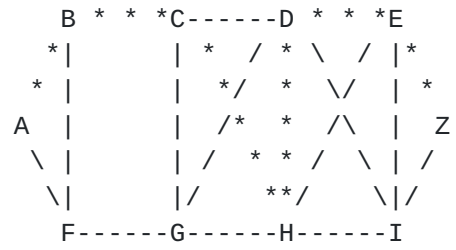

Figure 2: Bypass protection around link CD

When the primary path comes back up, the operator either allows for an automated reversion of the traffic onto the primary path or selects an operator-driven reversion.

3.2. Management-free Shortest Path Based Protection

An alternative protection strategy consists in management-free local protection, aiming at providing a repair for the destination based on the shortest path to the destination.

In our example, C protects Z, that it initially reaches via CD, by enforcing the traffic over its shortest path to Z, considering the failure of the protected component. The resulting end-to-end path between A and Z, upon recovery against the failure of CD, is depicted in Figure 3.

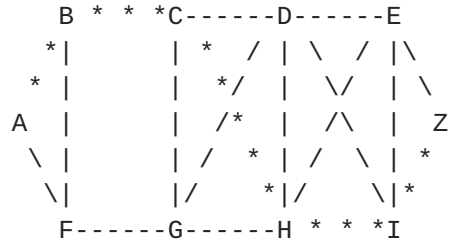

Figure 3: Shortest path protection around link CD

When the primary path comes back up, the operator either allows for an automated reversion of the traffic onto the primary path or selects an operator-driven reversion.

4. Managed Local Protection

There may be cases where a management free repair does not fit the policy of the operator. For example, in our illustration, the operator may not want to have CD and CH used to protect each other due the BW availability in each link and that could not suffice to absorb the other link traffic.

In this context, the protection mechanism MUST support the explicit configuration of the backup path either under the form of high-level constraints (end at the next-hop, end at the next-next-hop, minimize this metric, avoid this SRLG...) or under the form of an explicit path. Upon local detection of the failure, the traffic is repaired over the backup path in sub-50 milliseconds. When primary path comes back up, the operator either allows for an automated reversion of the traffic onto it or selects an operator-driven reversion.

We discuss such aspects for both bypass and shortest path based protection schemes.

4.1. Managed Bypass Protection

Let us illustrate the case using our reference example. For the demand from A to Z, the operator does not want to use the shortest failover path to the nexthop, {CH, HD}, but rather the path {CG, GH, HD}, as illustrated in Figure 4.

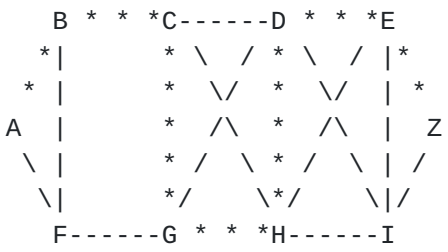

Figure 4: Managed Bypass Protection

The computation of the repair path SHOULD be possible in an automated fashion as well as statically expressed in the point of local repair.

4.2. Managed Shortest Path Protection

In the case of shortest path protection, the operator does not want to use the shortest failover via link CH, but rather reach H via {CG, GH}, for example, due to delay, BW, SRLG or other constraint.

The resulting end-to-end path upon activation of the protection is illustrated in Figure 5.

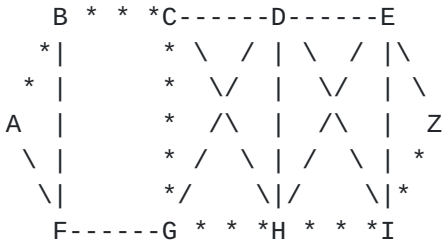

Figure 5: Managed Shortest Path Protection

The computation of the repair path SHOULD be possible in an automated fashion as well as statically expressed in the point of local repair.

The computation of the repair path based on a specific constraint SHOULD be possible on a per-destination prefix base.

5. Loop Avoidance

It is part of routing protocols behavior to have what are called "transient routing inconsistencies". This is due to the routing convergence that happens in each node at different times and during a different lapse of time.

These inconsistencies may cause routing loops that last the time that it takes for the node impacted by a network event to converge. These loops are called "microloops".

Usually, in a normal routing protocol operations, microloops do not last long and in general they are only noticed during the time it takes the network to converge. However, with the emerging of fast-convergence and fast-reroute technologies, microloops can be an issue in networks where sub-50 millisecond convergence/reroute is required. Therefore, the microloop problem needs to be addressed.

Networks may be affected by microloops during convergence depending of their topologies. Detecting microloops can be done during topology computation (e.g., SPF computation) and therefore microloops-avoidance techniques may be applied. An example of such technique is to compute microloop-free path that would be used during network convergence.

The SPRING architecture SHOULD provide solutions to prevent the occurrence of microloops during convergence following a change in the network state. Traditionally, the lack of packet steering capability made it difficult to apply efficient solutions to microloops. A SPRING enabled router could take advantage of the increased packet steering capabilities offered by SPRING in order to steer packets in a way that packets do not enter such loops.

6. Co-existence of multiple resilience techniques in the same infrastructure

The operator may want to support several very different services on the same packet-switching infrastructure. As a result, the SPRING architecture SHOULD allow for the co-existence of the different use cases listed in this document, in the same network.

Let us illustrate this with the following example:

- Flow F1 is supported over path {C, CD, E}
- Flow F2 is supported over path {C, CD, I}
- Flow F3 is supported over path {C, CD, Z}
- Flow F4 is supported over path {C, CD, Z}

It should be possible for the operator to configure the network to achieve path protection for F1, management free shortest path local protection for F2, managed protection over path {CG, GH, Z} for F3, and management free bypass protection for F4.

7. Security Considerations

This document describes requirements for the SPRING architecture to provide resiliency in SPRING networks. As such it does not introduce any new security considerations beyond that is discussed in [RFC7855].

8. IANA Considerations

This document does not request any IANA allocations.

9. Manageability Considerations

This document provides use cases. Solutions aimed at supporting these use cases should provide the necessary mechanisms in order to allow for manageability as described in [RFC7855].

Manageability concerns the computation, installation and troubleshooting of the repair path. Also, necessary mechanisms SHOULD be provided in order for the operator to control when a repair path is computed, how it has been computed and if it's installed and used.

10. Contributors

Pierre Francois contributed to the writing of the first version of this document.

11. Acknowledgements

Authors would like to thank Stephane Litkowski and Alexander Vainshtein for the comments and review of this document.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

12.2. Informative References

[RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <https://www.rfc-editor.org/info/rfc5286>.

[RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <https://www.rfc-editor.org/info/rfc5714>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

Authors' Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: stefano@previdi.net

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Rob Shakir
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043

Email: robjs@google.com