Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: April 30, 2009

M. Goyal
University of Wisconsin Milwaukee
G. Choudhury
AT&T
A. Shaikh
AT&T - Research
K. Trivedi
Duke University
H. Hosseini
University of Wisconsin Milwaukee

October 27, 2008

LSA Correlation to Schedule Routing Table Calculations
draft-goyal-ospf-lsacorr-00

Status of this Memo

Abstract

OSPF requires a router to recalculate its routing table whenever it receives a new router or network LSA. However, a topology change, such as a router down event, may cause a number of new LSAs to be generated. These LSAs may arrive at a router at different times. In order to avoid several routing table calculations in quick succession

in such cases, commercial routers enforce a hold time between successive routing table calculations. The hold time based schemes, while limiting the number of routing table calculations, may also cause undesirable delays in convergence to the topology change. This ID describes an alternate approach to schedule routing table calculations, called LSA Correlation. Rather than using individual LSAs as triggers for routing table calculations, LSA Correlation scheme correlates the information in the LSAs to identify the topology change. A routing table calculation can be performed when the topology change has been identified, which could span multiple LSAs.

Table of Contents

1. Introduction

OSPF requires a router to recalculate its routing table whenever it receives a new router or network LSA. However, a topology change, such as a router down event, may cause a number of new LSAs to be generated. These LSAs may arrive at a router at different times. In order to avoid several routing table calculations in quick succession in such cases, commercial routers enforce a hold time between successive routing table calculations. The hold time based schemes, while limiting the number of routing table calculations, may also cause undesirable delays in convergence to the topology change. This ID describes an alternate approach to schedule routing table calculations, called LSA Correlation. Rather than using individual LSAs as triggers for routing table calculations, LSA Correlation scheme correlates the information in the LSAs to identify the topology change. A routing table calculation can be performed when the topology change has been identified, which could span multiple LSAs.

The proposed LSA correlation scheme is based on the fact that a new LSA is just a symptom of a topology change and hence should not be used as the trigger for a routing table calculation. Rather, a router should correlate the information in individual new LSAs to identify the topology change itself and then perform a routing table calculation.

2. 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].

3. LSA Correlation

In the following, we discuss how to correlate the individual router and network LSAs to identify the topology change(s). In this discussion, we have used the term 'node' to refer to both a 'router' and a 'transit network'. Also, any reference to scheduling an 'immediate' routing table calculation means that a routing table calculation is performed after completing the processing of the current 'Link State Update' packet.

The LSA correlation process consists of the following steps.

- Step 1: Identify an 'up', 'down' or 'cost change' subevent by iterating through the contents of the new LSA and its old version.

This step has O(k) running time complexity, where 'k' is the number of link states contained in the LSA.

- Step 2: Correlate the subevents to identify a topology change. This step has O(1) running time complexity for each subevent.

Every new router and network LSA needs to undergo this correlation task with an overall running time complexity O(k). Here, 'k' corresponds to the number of neighbors (routers and networks) of the node originating the LSA, which is typically a small number. As discussed later, the identification of a "node down" event requires some additional processing with running time complexity O(k). Note that the OSPFv2 specification [RFC2328] requires the new instance of an LSA to be compared to the old instance to determine if a routing table calculation is required. Hence, many OSPF implementations may already be doing most of the processing required for LSA correlation. Thus, the additional processing overhead of LSA Correlation procedure should be insignificant.

Link Type	Link States (LS) in an LSA
Point-to-point link	one type 3 LS
	one type 1 LS (after adj establishment)
Broadcast/NBMA link	one type 3 LS (before adj est.)
	one type 2 LS (after adj est.)
Point-to-multipoint link	one type 3 LS
	multiple type 1 LS (after adj est.)
Virtual link	one type 4 link state

Table 1: Different Link Types in OSPF and the Corresponding Link States in an LSA

3.1. How to identify a topology change?

In the following, we list typical topology changes and the criteria for their identification.

- Link Down: A link can be declared \emph{down} if either end breaks adjacency with the other.

- Link Up: A link can be declared \emph{up} if both ends establish adjacency with each other.

- Node Down: A node can be declared \emph{down} if no node is currently adjacent to it.

- Node Up: A node can be declared \emph{up} if it has established adjacency with all its known neighbors and all the neighbors have also established adjacency with this node.

3.1.1. Identifying link/node down events

A "link down" event could be a part of a "node down" event. In case of a "link down" event, both ends of the link will break adjacency with each other and hence both ends will generate new LSAs. But for a "node down" event, the down node will not generate a new LSA. Hence, one way to distinguish a "link down" event from a "node down" event is to wait for new LSAs from both ends of the link announcing the breakdown of adjacency with each other. However, such a wait will delay the convergence to the "link down" events, which constitute the most common case of network failures.

We distinguish between "link down" and "node down" events as follows. Consider two nodes 'A' and 'B' that are currently adjacent to each other. Suppose no node has recently broken adjacency with node 'B'. Further, suppose that node 'A' generates a new LSA that no longer indicates an adjacency with node 'B'. Now, the topology change that led to the generation of this LSA could be the failure of link 'A:B' or the failure of node 'B' itself. In any case, we schedule an immediate routing table calculation, thereby ensuring quick convergence to a possible "link 'A:B' down" event. To prepare for the possibility that node 'B' is down, we mark node 'B' as in the process of "going down" and also decrement the number of bidirectional adjacencies of both nodes 'A' and 'B'.

If node 'B' indeed went down, its other neighbors would also soon generate LSAs indicating break down of their adjacency with node 'B'. Suppose, we receive one such LSA from node 'C' showing that it is no longer adjacent with node 'B'. This time, since node 'B' is marked as "going down", a routing table calculation is not performed. Rather, we decrement the count of the bidirectional adjacencies associated with nodes 'B' and 'C' and wait for other nodes currently adjacent to node 'B' to break their adjacency too. We also (re)start a timer, called "doSPF". The purpose of the "doSPF" timer is to assimilate all the pending LSAs into the routing table if the topology change(s) can not be identified before the timer's firing. The firing of this timer results in a routing table calculation. The timer is stopped if a routing table calculation is performed while the timer is running. Once no node is adjacent to node 'B' any more, we schedule an immediate routing table calculation.

On the other hand, if node 'B' is not down, it will soon generate a new LSA announcing the break down of its adjacency with node 'A'. This LSA will cause node B's "going down" marking to be undone. This

solution ensures that convergence to "link/node down" events is quick. A "link down" event requires one routing table update while a "node down" event requires two. In case node 'B' is down but it is not possible to receive adjacency breakdown indications from all its currently adjacent neighbors (because they are down too or they can no longer be reached), a routing table calculation will be performed when the "doSPF" timer fires.

On the identification of a "node down" event, all the adjacencies and stub networks associated with the node are marked as down, which requires iterating through the last LSA received from the down node and has a time complexity of O(k), where 'k' is the number of link states in the LSA. This is the additional processing required, alluded to earlier, following the identification of a "node down" event.

3.1.2. Identifying link/node up events

A "link up" event could be distinguished from a "node up" event by checking whether both ends of the link are already "up" or not. Both ends of the link would be "up" only in case of a pure "link up" event. If either end of the link is not currently "up", the "link up" event must be a part of a "node up" event. A node can be declared to be "up" when it has established adjacency with all its known neighbors, i.e., when the number of bidirectional adjacencies for the node is same as the number of its neighboring nodes.

Suppose node 'A' generates an LSA that indicates a new adjacency with node 'B'. If the last LSA from node 'B' did not indicate node 'A' as adjacent, there is nothing to be done. Otherwise, we increment the number of bidirectional adjacencies for nodes 'A' and 'B'. If both nodes 'A' and 'B' are currently considered "up", we declare link 'A:B' as "up". Otherwise, we consider this adjacency establishment as part of a node "up" event. If either node 'A' or node 'B' is currently considered to be down, we check if its bidirectional adjacency count is equal to the number of its known neighbors and if so, we declare the node to be "up".

The number of neighbors for a node can be determined by examining the node's LSA. Table Table 1 shows different types of OSPF links and the information contained in an LSA for each link type in OSPF version 2. Clearly, "max(type 3 link states, type 1/2/4 link states)" gives the number of neighbors for the node originating the LSA. We count only bidirectional adjacencies in the test for "node up" event since, in OSPF, the routing table calculation uses only those links where bidirectional adjacency exists between the two ends.

To deal with the possibility that a newly up node may not establish adjacency with all its neighbors, we (re)start the "doSPF" timer when a new bidirectional adjacency is identified. The expiry of the "doSPF" timer will cause a routing table calculation, thereby assimilating all the new LSAs received so far into the routing table.

The scheme, described above, can be referred to as "simple" LSA correlation.

3.2. Avoiding multiple routing table calculations for multiple concurrent topology changes

There are several scenarios where multiple topology changes take place concurrently or in quick succession. For example, a cut in an optical fiber would cause failure of all the IP links it carries. An optical fiber is an example of a "shared risk link group" (SRLG), which is defined as a group of links that share a common risk of failure, i.e., all the links in the SRLG will fail if the risk materializes. Another example is a "point of presence" (PoP), which refers to a physical location where an Internet Service Provider (ISP) locates the networking hardware such as routers. All the routers in a PoP may fail together if a power failure affects the entire PoP. If multiple topology changes occur in quick succession, doing a new routing table calculation immediately on identifying a topology change could lead to several back-to-back calculations in the routers.

In order to avoid multiple routing table calculations in case of multiple concurrent "node up/down" events, we propose the following modification in the LSA correlation process: do not perform a routing table calculation as long as some nodes are in the process of "going down" or "coming up". As discussed earlier, a node is marked as "going down" if atleast one neighbor has recently broken adjacency with it. We need to additionally keep track of the nodes that are in the process of "coming up". We can infer that a node is in the processing of "coming up" if it originates a new LSA but has not established adjacency with all its neighbors. When a node has established adjacency with all its neighbors (i.e. is declared up), its "coming up" marking is undone. Whenever a link/node "up" or "down" event is identified, we schedule an immediate routing table calculation only if no node is currently marked as "coming up" or "going down".

To protect against pathological situations such as "link flaps" that may keep one or more nodes in "coming up" or "going down" states forever, a "pending" timer is (re)started every time a routing table calculation is postponed in this manner. If some nodes are still in the process of "coming up" or "going down" when this timer fires, a

routing table calculation is performed to assimilate all recent topology changes into the routing table. Note that this optimization does not impact fast convergance to "isolated" link/node up/down events. This scheme is referred to as "conservative" LSA correlation.

Another optimization is to avoid routing table calculations while a router is establishing adjacency with a neighbor. The correlation of the LSAs received during the link state database exchange may cause identification of several "topology changes". Thus, without this optimization, a router may end up doing several routing table calculations while establishing adjacency with a neighbor. Once the adjacency establishment process is over, the two newly adjacent routers will generate new LSAs, which when correlated will cause a routing table calculation to take place.

Note that multiple routing table calculations in case of multiple concurrent topology changes can also be avoided using a fixed/ exponential hold time based scheme. In such a scheme, individual topology changes, identified via LSA correlation, could serve as the triggers for driving the scheme's state machine. Further, it is possible to identify and handle pathological conditions such as link flaps using the "up/down" subevents generated during the LSA correlation process. An implementation of the LSA correlation process in a modified "ospfd" simulator is publically available [newospfd].

4. IANA Considerations

This memo includes no request to IANA.

5. Security Considerations

TBD

6. References

6.1. Normative References

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

6.2. Informative References

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[newospfd] Goyal, M., "A distributed OSPFD simulator", 2006, <http://www.cs.uwm.edu/~mukul/newospfd.html>.

Authors' Addresses

Mukul Goyal
University of Wisconsin Milwaukee
3200 N Cramer St
Milwaukee, WI 53201
USA

Phone: +1 414 229 5001
Email: mukul@uwm.edu

Gagan Choudhury
AT&T
200 Laurel Avenue
Middletown, NJ 07748
USA

Phone: +1 732 420 3721
Email: gchoudhury@att.com

Aman Shaikh
AT&T - Research
180 Park Avenue
Florham Park, NJ 07932
USA

Phone: +1 973 360 7288
Email: ashaikh@research.att.com

Kishor Trivedi
Duke University
Durham, NC 27708
USA

Phone: +1 919 660 5269
Email: kst@ee.duke.edu

Hossein Hosseini
University of Wisconsin Milwaukee
3200 N Cramer St
Milwaukee, WI 53201
USA

Phone: +1 414 229 5184
Email: hosseini@uwm.edu

Full Copyright Statement

Intellectual Property