Abstract

OSPF [RFC2328] is a link-state intra-domain routing protocol used in IP networks. OSPF behavior over demand circuits is optimized in [RFC1793] to minimize the amount of overhead traffic. A part of OSPF demand circuit extensions is the Hello suppression mechanism. This technique allows a demand circuit to go down when no interesting traffic is going through the link. However, it also introduces a problem, where it becomes impossible to detect an OSPF-inactive neighbor over such a link. This memo addresses the above problem by the neighbor probing mechanism.

1. Motivation

In some situations, when operating over demand circuits, the remote
neighbor may be unable to run OSPF, and, as a possible result, unable to route application traffic. Possible scenarios include:

- The OSPF process might have died on the remote neighbor.
- Oversubscription (Section 7 of [RFC1793]) may cause a continuous drop of application data at the link level.

The problem here is that the local router cannot identify the problems such as this, since Hello exchange is suppressed on demand circuits. If the topology of the network is such that other routers cannot communicate their knowledge about the remote neighbor via flooding, the local router and all routers behind it will never know about the problem, so application traffic may continue being forwarded to the OSPF-incapable router.

This memo describes a backward-compatible neighbor probing mechanism based on the details of the standard flooding procedure followed by OSPF routers.

2. Proposed Solution

The solution this document proposes uses LSA update packets to detect whether the OSPF process is operational on the remote neighbor. We call this process "Neighbor probing". The idea behind this technique is to allow either of the two neighbors connected over a demand circuit to test the remote neighbor at any time (see Section 2.1).

The routers across the demand circuit can be connected by either a point-to-point link, or a virtual link, or a point-to-multipoint interface. The case of routers connected by broadcast networks or NBMA is not considered, since Hello suppression is not used in these cases (Section 3.2 [RFC1793]).

The neighbor probing mechanism is used as follows. After a router has synchronized the LSDB with its neighbor over the demand circuit, the demand circuit may be torn down if there is no more application traffic. When application traffic starts going over the link, the link is brought up, and the routers may probe each other. The routers may also periodically probe each other any time the link is up (could be implemented as a configurable option) with the caution that OSPF packets sent as a part of neighbor probing are not considered as interesting traffic and do not cause the demand circuit to remain up (relevant details of implementation are outside of the scope of this document).

The case when one or more of the router’s links are oversubscribed (see section 7 of [RFC1793]) should be considered by the
implementations. In such a situation even if the link status is up and application data being sent on the link, only a limited number of neighbors is really reachable. To make sure temporarily unreachable neighbors are not mistakenly declared down, Neighbor probing should be restricted to those neighbors that are actually reachable (i.e., there is a circuit established with the neighbor at the moment the probing procedure needs to be initiated). This check itself is also considered an implementation detail.

2.1 Neighbor Probing

The neighbor probing method described in this section is completely compatible with standard OSPF implementations, because it is based on standard behavior that must be followed by OSPF implementations in order to keep their LSDBs synchronized.

When a router needs to verify OSPF capability of a neighbor reachable through a demand circuit, it should flood to the neighbor any LSA in its LSDB that would normally be sent to the neighbor during the initial LSDB synchronization process (it most cases such LSA must have already been flooded to the neighbor by the time the probing procedure starts). For example, the router may flood its own router-LSA (without originating a new version), or the neighbor’s own router-LSA. If the neighbor is still alive and OSPF-capable, it replies with a link state acknowledgement or a link state update (an implied acknowledgement) and the LSA is removed from the neighbor’s retransmission list. The implementations should limit the number of times an LSA can be retransmitted when used for neighbor probing. If no acknowledgement (explicit or implicit) is received for a predefined period of time, the probing router should treat this as evidence of the neighbor’s unreachability (proving wrong the assumption of reachability used in [RFC1793]) and should bring the adjacency down.

Note that when the neighbor being probed receives such a link state update packet, it acknowledges the LSA but does not flood it any further since received copy of the LSA is considered to be the same as the neighbor’s database copy. Because of this property, the link state update based neighbor probing mechanism is localized to the demand circuit and does not increase flooding in the area.

Again, the implementation should insure (through internal mechanisms) that OSPF link state update packets sent over the demand circuit for the purpose of neighbor probing do not prevent that circuit from being torn down.

3. Support of Virtual Links and Point-to-multipoint Interfaces
Virtual links can be treated analogous to point-to-point links and so the techniques described in this memo are applicable to virtual links as well. The case of point-to-multipoint interface running as demand circuit (Section 3.5 [RFC1793]) can be treated as individual point-to-point links, for which the solution has been described in Section 2.

4. Compatibility issues

All mechanisms described in this document are backward-compatible with standard OSPF implementations.

5. Considerations

In addition to the lost functionality mentioned in Section 6 of [RFC1793], there is an added overhead in terms of the amount of data (link state updates and acknowledgements) being transmitted due to neighbor probing whenever the link is up and thereby increasing the overall cost.

6. Acknowledgements

The authors would like to thank John Moy, Vijayapal Reddy Patil, SVR Anand, and Peter Psenak for their comments on this work.

A significant portion of Sira’s work was carried out as part of the HFCL-IISc Research Project (HIRP), Bangalore, India. He would like to thank the team for their insightful discussions.

7. References

[RFC2328]

[RFC1793]

8. Authors’ addresses

Sira Panduranga Rao                      Alex Zinin
The University of Texas at Arlington     Nexsi Systems
Arlington, TX 76013                      1959 Concourse Drive
Email: siraprao@hotmail.com              San Jose, CA 95131
                                           Email: azinin@nexsi.com