Network Working Group
Internet Draft
Expiration Date: September 2001
File name: draft-ietf-ospf-isis-flood-opt-01.txt

Alex Zinin
Mike Shand
Cisco Systems
March 2001

Flooding optimizations in link-state routing protocols

draft-ietf-ospf-isis-flood-opt-01.txt

Status of this Memo

Abstract

The flooding algorithm is one of the most important parts of any link state routing protocol. It ensures that all routers within a link state domain converge on the same topological information within a finite period of time. To ensure reliability, typical implementations of the flooding algorithm send new information via all interfaces other than the one the new piece of information was received on. This redundancy is necessary to guarantee that flooding is performed reliably, but implies considerable overhead of utilized bandwidth and CPU time if neighboring routers are connected with more than one link. This document describes a method that reduces this overhead.

1 Introduction

In order to guarantee convergence of a link state routing protocol, it is vital to ensure that link state PDUs (LSAs in the case of OSPF [Ref1] or LSPs in the case of ISIS [Ref2]) that are originated after the initial LSDB synchronization between neighbors is completed are delivered to all routers within the flooding scope limits (an area or the whole AS depending on the protocol and the type of the link state PDU).

The method used by link state protocols to achieve this implies that a) PDUs are transmitted reliably between any pair of routers, and b) whenever a new PDU is received, it is sent across all interfaces other than the one it was received on (except for the case when the router is the DR in OSPF, where the LSA is sent back over the same interfaces, see [Ref1]).

To fulfil the first requirement, link state routing protocols keep retransmitting new PDUs to the neighbors that have not acknowledged reception (the only exception is flooding performed on broadcast links in ISIS, see [Ref2]). As an example, in OSPF, a link state retransmission list is maintained for every neighbor data structure on every interface. When an LSA is sent through an interface, it is put on the retransmission list of every neighbor associated with this interface and is removed from it only after the neighbor has acknowledged reception of the LSA. Similarly, ISIS implementations use SRMflag and SSNflag that are interface-specific, as well as periodical CSNP announcements on broadcast links to ensure reliability of flooding.

2 Motivation

If multiple links connect two routers, flooding of new information will cause considerable overhead of link bandwidth and CPU time spent by the protocol. Let's consider an example shown in Figure 1.

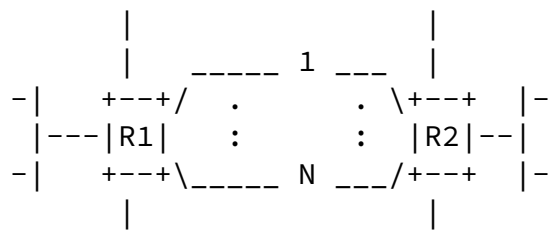

Figure 1. Sample topology

When R1 receives a new PDU from its LAN segment, it installs it in

its LSDB and submits for flooding through all of its interfaces. Since flooding presumes sending the new PDU over all interfaces except for the one it was received on routers end up doing the following.

1) R1 sends not one, but N copies of the new PDU to R2.

2) Only the first copy of the PDU is actually installed in R2's LSDB, but link bandwidth and CPU cycles are spend to transmit and process all N copies.

3) Furthermore, when R2 receives the first copy of the LSA and installs it, it floods back to R1 N-1 copies of it, again spending extra bandwidth and CPU time.

4) If R1 receives an acknowledgment from R2 on some links, but not from others, it will keep retransmitting unacknowledged LSAs though they are already in R2's LSDB.

The solution described in this document provides a technique to minimize the overhead that link state routing protocols cause in the described situation and use link bandwidth more efficiently.

While the described optimization is generic for OSPF and ISIS, it becomes very important in the contents of MPLambdaS [Ref3] where OXCs may be connected with a large number of links. The problem is partially addresses by LMP [Ref4] where a single control channel is used for a bundle of optical links or lambdas. However, OXCs may still have a big amount of control channels between each other, and some type of OXCs may not be running LMP at all. The flooding optimizations described in this document ensure scalability of the IGP flooding algorithm in the presence of multiple links between neighboring routers and increasing amount of traffic engineering information flooded in optical and MPLS networks.

3 Proposed solution

3.1 Introduction

The main idea of the technique described in this document is to move the flooding algorithm from the per-interface to per-neighbor basis in a backward-compatible manner.

The technique is generic for all protocols utilizing reliable flooding and is based on the observation that the ultimate goal of the flooding algorithm is not to send link state PDUs over all interfaces, but to deliver them to all routers in the network.

To implement this optimization, it is necessary to maintain a list of neighbors within an area. Whenever a new neighbor is discovered on an interface belonging to the area, the corresponding interface neighbor data structure is linked to the corresponding element in the list of neighbors. Based on the information in the list of neighbors, as well as on the type of interfaces they use, interfaces within the area are marked either flooding-active or flooding-passive. The process of election of flooding-active interfaces takes into consideration the costs of interfaces, giving preference to faster interfaces. Multiaccess interfaces need special treatment, since they may be (usually are) associated with more than one neighbor. However, if such an interface connects only two routers, it still may be marked as flooding-passive. Whenever the number of entries in the list or state of the adjacency in the list changes, the interface election algorithm is rerun.

Note that since the flooding paradigm is changed from the per-interface to per-neighbor basis, PDU retransmission is not performed for a specific neighbor on a specific interface, but is instead done for a specific neighbor in general, and it is enough to receive a single acknowledgment on any interface for sending router to stop retransmitting.

The asynchronous flooding algorithm is changed to first consider the area neighbor list and then use available physical interfaces to reliably deliver link state PDUs to the neighbors. Note that if more than one interface to a particular neighbor is marked as flooding-active, the flooding algorithm may perform equal- or unequal-cost load sharing, flooding different PDUs through different links.

Flooding is also changed not to send PDUs to the sending neighbor via other links.

The initial process of LSDB synchronization is also changed to take advantage of multiple links. If a new adjacency is coming up and the router can be sure that its LSDB is already synchronized with the remote router over other links, the router can speed up the adjacency establishment process by sending a empty (or limited-size) database description to the remote neighbor. Note that this speeds up the announcement of links that come up, since OSPF announces an adjacency only when it reaches Full state, i.e., when routers have synchronized their LSDBs. Skipping the LSDB synchronization part in ISIS does not speed link announcement (since ISIS announces adjacency as soon as two-way connectivity has been ensured), but it reduces the amount of time CPU spends on processing of CSNPs and LSPs.

To illustrate the benefits of the described method, consider the situation where R1 in Figure 1 has 100 PDUs to flood to R2, and N

equals 3. Without the described optimization, we would have:

- 300 copies of PDUs going from R1 to R2
- 300 LSDB lookups performed by R2
- 300 acknowledgements coming back from R2
- 300 lookups on the retransmit list or LSDB by R1 to remove acknowledged PDUs
- 200 copies of PDUs coming back from R2 to R1
- 200 LSDB lookups performed by R1
- 200 ackowledgments going from R1 to R2
- 200 lookups on the retransmit list or LSDB by R2 to remove acknowledged PDUs

If described technique is implemented, we would have:

- 100 copies of PDUs going from R1 to R2 possibly over different interfaces for faster transmission
- 100 LSDB lookups performed by R2
- 100 acknowledgements coming back from R2
- 100 lookups on the retransmit list or LSDB by R1 to remove acknowledged PDUs
- no copies of PDUs coming back from R2 to R1
- no LSDB lookups performed by R1
- no acknowledgments going from R1 to R2
- no lookups on the retransmit list or LSDB by R2 to remove acknowledged PDUs

3.2 Changes to OSPF

3.2.1 Data structures

Some basic modifications to OSPF data structures are necessary to implement the described solution.

First of all, a new field is introduced to the area data structure, called NeighborList. NeighborList is a list of entries, each containing the following fields. Note that the NeighborList entry is created when the first neighbor data structure for the neighbor with a particular router ID is created within an area.

- NeighborID---the router ID of the neighbor connected with the calculating router with one or more interfaces.
- P2pIntList---list of interfaces that have only one fully established adjacency and it is established with the neighbor identified by the NeighborID (point-to-point and virtual links, as well as broadcast and NBMA interfaces connecting only two routers).

- o P2mpIntList---list of interfaces that have more than one fully established adjacency and one of them is established with the neighbor identified by the NeighborID (apparently NBMA and broadcast networks).

- o Retransmission list---list of LSAs that must be delivered to the remote router using available physical interfaces.

Note that when a neighbor is reachable over multiple interfaces, there will be more than one entry in the above lists of interfaces.

A new field is introduced to the interface-specific neighbor data structure---NeighborEntry. When an instance of the interface-specific neighbor data structure is created, its NeighborEntry is set to reference the corresponding entry in the area neighbor list. Note that whenever a neighbor data structure is created for an interface, the P2pIntList and P2mpIntList of area neighbor data structures corresponding to the neighbors reachable through the same interface are modified. If only one neighbor data structure is available for the interface, the interface is put on the P2pIntList for that neighbor. Otherwise, if more than one neighbor is known (regardless of the state of the neighbors), the interface is placed on the P2mpIntLists of all neighbors reachable through that interface. Note that point-to-point interfaces may temporarily have more than one neighbor linked to the interface data structure when the router-ID of the neighbor is changing. The algorithm handles such transition states by temporarily putting the interface on the P2mpIntList.

Also, a new field is introduced to the interface data structure, called FloodingActive. If the value of this field is TRUE, the interface is used for flooding. Otherwise the interface is flooding- passive and no LSAs are sent over it when asynchronous flooding is performed. Note that whenever an interface is put on a P2mpIntList of

any area neighbor data structure, its FloodingActive field is always set to TRUE. Actually, FloodingActive field is consulted only if the interface is in a P2pIntList.

Another field introduced to the interface data structure is LSASent, that is used by the flooding procedure (see Section 3.2.3 for more details).

Whenever there is a change in the contents of the P2pIntList or P2mpIntList of an area neighbor data structure, the router performs election of flooding-active interfaces among the interfaces listed in the P2pIntList field.

Below follows the algorithm describing the election process. Note that this algorithm produces the minimal set of active interfaces. Implementations may use different algorithms, but these algorithms must not produce a smaller set of interfaces.

For every entry in the area neighbor list, do the following.

1. If the P2mpIntList is not empty, go through all interfaces in the P2pIntList and mark them flooding-passive by setting the FloodingActive interface field to FALSE. (We always prefer sending LSAs to multiple neighbors simultaneously).

2. Otherwise, among the interfaces in the P2pIntList, set FloodingActive field to TRUE for those interfaces that have the best interface cost. Set it to FALSE for all other interfaces in the list.

3.2.2 Initial LSDB synchronization

Implementations may decide to maintain a single link state request list per neighbor in an area. This may be used to split the Loading process among several links when more than one adjacency is coming up simultaneously. Note that in this case, whenever the link state request list for a particular neighbor becomes empty, a LoadingDone event should be generated for all adjacencies with this neighbor that are currently in the Loading state.

3.2.3 Asynchronous Flooding

Asynchronous flooding algorithm is changed as follows. Note that changes described below do not affect flooding back to a multiaccess interface if the router is the DR. The changes are only in the part where LSA is sent over other interfaces.

If the flooding scope is domain-wide, perform the following for

all areas. If the flooding scope is area-wide, do the following

steps only for the area the interface on which the LSA was received belongs to.

Consider every neighbor element in the area neighbor list as follows.

1) If the value of the NeighborID field is equal to the router ID of neighbor that sent the LSA to the router, consider the next neighbor element (there is no need to send the LSA back to the sending router, except for the case when the receiving router is the DR and the LSA is flooded back to a mutliaccess interface).

2) If P2mpIntList is empty, go to step 3. Otherwise do the following steps

   a) Put the LSA on the neighbor's retransmission list

   b) Go through every interface on the P2mpIntList and do the following:

      - Compare the LSA being flooded and the one identified by the LSASent field of the interface data structure. If the LSAs are the same, the LSA has already been sent on this interfaces and next interface in P2mpIntList must be considered.

      - Send the LSA in a link state update packets setting the destination address according to the rules in Section 13 of [Ref1]

      - Set LSASent field of the interface data structure to the LSA that has just been sent.

   c) Consider the next neighbor element in the area neighbor list. (That is, skip flooding over interfaces in the P2pIntList.)

3) If P2pIntList is empty, consider the next neighbor element. Otherwise:

   a) Put the LSA on the neighbor's retransmission list

   b) Go through every interface on the P2pIntList and do the following:

      - If the interface FloodingActive flag is clear,

skip this interface and consider the next interface in the list.

- o Send the LSA in a link state update packets setting the destination address according to the rules in Section 13 of [Ref1]

c) Consider the next neighbor element in the area neighbor list.

Reception of OSPF acknowledgements is modified as follows.

Whenever a link state acknowledgement is received from a neighbor, the corresponding entry in the area neighbor list is located and corresponding LSA is removed from the retransmission list.

3.2.4 Retransmitting LSAs

The OSPF implementation should also be modified to perform retransmission of LSAs on a per-neighbor basis.

Normally, the interfaces for LSA retransmission should be selected according to the rules used for asynchronous LSA flooding. However, implementations may consider retransmitting LSAs over a bigger set of interfaces leading to the neighbor if the minimal interface set is suspected to be not sufficient (because of link load, or packet drops) to complete LSDB synchronization within a reasonable period of time.

3.2.5 Opaque LSA Support

Described optimizations can be applied to all LSAs, including Opaque LSAs that have area and domain wide flooding scope (type-10 and type-11). Note however, that transmission of type-9 LSAs (that have link-local flooding scope) should remain intact.

3.2.6 Compatibility

The optimization described above is designed to be backward-compatible. No software modification is necessary for the neighboring routers. However, if both routers support the described modifications, the advantages will be greater.

3.3 Changes to ISIS

The changes for IS-IS are similar to those for OSPF.

Each non-broadcast circuit has associated with it the system ID of

the neighbor that is adjacent over that circuit. At each of level 1 and level 2, the set of one or more circuits with an adjacency at that level and a common neighbor is identified as a group. SRMflags are associated with groups rather than circuit. SSNflags remain associated with circuits.

ISO/IEC 10589 describes the setting or clearing of SRMflag or SSNflag on a non-broadcast circuit for the following reasons.

1. SSNflag is cleared after the transmission of a PSNP over circuit C.

2. A packet has been received on circuit C and a flag on circuit C set or cleared as a result.

3. A packet has been received on circuit C and the flags on all other circuits set or cleared as a result.

4. The flags on all circuits are set or cleared.

These actions are modified as described below. In these descriptions, the term Sxxflag refers to either SSNflag or SRMflag.

1. SSNflag is cleared after the transmission of a PSNP over circuit C.

   Clear the flag on circuit C.

2. A packet has been received on circuit C and an Sxxflag on circuit C is to be set or cleared as a result. Circuit C is a member of group G.

   a. If an SSNflag is to be cleared, clear ALL SSNflags for circuits in group G.

   b. If an SRMflag is to be cleared, clear the SRMflag for group G.

   c. If an SSNflag is to be set, set the SSNflag for circuit C

only.

d. If an SRMflag is to be set, set the SRMflag for group G.

3. A packet has been received on circuit C and the Sxxflags on all other circuits are to be set or cleared as a result. Circuit C is a member of group G.

a. If an SSNflag is to be cleared, clear ALL SSNflags for

all circuits belonging to groups other than G.

b. If an SRMflag is to be cleared, clear ALL SRMflags for groups other than G.

c. If an SRMflag is to be set, set ALL SRMflags for groups other than G.

4. The flags on all circuits are set or cleared

a. If an SSNflag is to be cleared, clear ALL SSNflags for all circuits.

b. If an SRMflag is to be cleared, clear SRMflags for all groups.

c. If an SRMflag is to be set, set SRMflags for all groups.

5 Transmitting an LSP as a result of SRMflags being set on group G.

Choose ONE circuit from group G, and transmit the LSP over that circuit.

Where a circuit is required to be chosen from within a group, the choice made is implementation dependant and may be based on any criteria, such as bandwidth or management control. The result of the choice MAY be different on each occasion. Implementations may also decide to choose no point-to-point links if a neighboring system is available via a broadcast circuit, since LSPs need to be flooded throught it anyway. It is also possible to treat broadcast circuits with only two routers attached as point-to-point circuits (inlcuding

Hello PDUs and LSDB synchronization pocess), note however that the routers should be explicitly configured to do so.

4 Security issues

This document does not introduce any new security issues to ISIS or OSPF.

5 Acknowledgements

The authors would like to acknowledge Tony Przygienda, Yakov Rekhter, and John Moy for their comments, and Jeff Learman for reviewing this document.

6 References

[Ref1] J. Moy. OSPF version 2. Technical Report RFC 2328, Internet Engineering Task Force, 1998. ftp://ftp.isi.edu/in-notes/rfc2328.txt.

[Ref2] ISO, "Intermediate system to Intermediate system routeing information exchange protocol for use in conjunction with the Protocol for providing the Connectionless-mode Network Service (ISO 8473)," ISO/IEC 10589:1992.

[Ref3] D. Awduche, Y. Rekhter, J. Drake, R. Coltun, "Multi-Protocol Lambda Switching: Combining MPLS Traffic Engineering Control With Optical Crossconnects", draft-awduche-mpls-te-optical-02.txt. Work in progress.

[Ref4] J. Lang, et al. "Link Management Protocol (LMP)", draft-ietf-mpls-lmp-01.txt. Work in progress.

7 Authors' addresses

Alex Zinin
Cisco Systems
150 West Tasman Dr.
San Jose,CA
95134
E-mail: azinin@cisco.com

Mike Shand
Cisco Systems
4, The Square
Stockley Park
UXBRIDGE
Middlesex

UB11 1BN, UK
E-mail: mshand@cisco.com