Network Working Group
Request for Comments: 2642
Category: Informational

L. Kane
Cabletron Systems Incorporated

August 1999

Cabletron's VLS Protocol Specification

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The Virtual LAN Link State Protocol (VLSP) is part of the InterSwitch Message Protocol (ISMP) which provides interswitch communication between switches running Cabletron's SecureFast VLAN (SFVLAN) product. VLSP is used to determine and maintain a fully connected mesh topology graph of the switch fabric. Each switch maintains an identical database describing the topology. Call-originating switches use the topology database to determine the path over which to route a call connection.

VLSP provides support for equal-cost multipath routing, and recalculates routes quickly in the face of topological changes, utilizing a minimum of routing protocol traffic.

Table of Contents

1. Introduction

This memo is being distributed to members of the Internet community in order to solicit reactions to the proposals contained herein. While the specification discussed here may not be directly relevant to the research problems of the Internet, it may be of interest to researchers and implementers.

1.1 Acknowledgments

VLSP is derived from the OSPF link-state routing protocol described in [RFC2328], written by John Moy, formerly of Proteon, Inc., Westborough, Massachusetts. Much of the current memo has been drawn from [RFC2328]. Therefore, this author wishes to acknowledge the contribution Mr. Moy has (unknowingly) made to this document.

1.2 Data Conventions

The methods used in this memo to describe and picture data adhere to the standards of Internet Protocol documentation [RFC1700]. In particular:

The convention in the documentation of Internet Protocols is to express numbers in decimal and to picture data in "big-endian" order. That is, fields are described left to right, with the most significant octet on the left and the least significant octet on the right. The order of transmission of the header and data described in this document is resolved to the octet level. Whenever a diagram shows a group of octets, the order of transmission of those octets is the normal order in which they are read in English.

Whenever an octet represents a numeric quantity the left most bit in the diagram is the high order or most significant bit. That is, the bit labeled 0 is the most significant bit.

Similarly, whenever a multi-octet field represents a numeric quantity the left most bit of the whole field is the most significant bit. When a multi-octet quantity is transmitted the most significant octet is transmitted first.

1.3 ISMP Overview

The InterSwitch Message Protocol (ISMP) provides a consistent method of encapsulating and transmitting control messages exchanged between switches running Cabletron's SecureFast VLAN (SFVLAN) product, as described in [IDsfvlan]. ISMP provides the following services:

- Topology services. Each switch maintains a distributed topology of the switch fabric by exchanging the following interswitch control messages with other switches:

- Interswitch Keepalive messages are sent by each switch to announce its existence to its neighboring switches and to establish the topology of the switch fabric. (Interswitch Keepalive messages are exchanged in accordance with Cabletron's VlanHello protocol, described in [IDhello].)

- Interswitch Spanning Tree BPDU messages and Interswitch Remote Blocking messages are used to determine and maintain a loop-free flood path between all network switches in the fabric. This flood

  path is used for all undirected interswitch messages -- that is, messages that are (potentially) sent to all switches in the switch fabric.

- Interswitch Link State messages (VLS protocol) are used to determine and maintain a fully connected mesh topology graph of the switch fabric. Call-originating switches use the topology graph to determine the path over which to route a call connection.

- Address resolution services. Interswitch Resolve messages are used to resolve a packet destination address when the packet source and destination pair does not match a known connection. Interswitch New User messages are used to provide end-station address mobility between switches.

- Tag-based flooding. A tag-based broadcast method is used to restrict the broadcast of unresolved packets to only those ports within the fabric that belong to the same VLAN as the source.

- Call tapping services. Interswitch Tap messages are used to monitor traffic moving between two end stations. Traffic can be monitored in one or both directions along the connection path.

Note: Previous versions of VLSP treated all links as if they were broadcast (multi-access). Thus, if VLSP determines that a neighbor switch is running an older version of the protocol software (see Section 6.1), it will change the interface type to broadcast and begin exchanging Hello packets with the single neighbor switch.

2. VLS Protocol Overview

VLSP is a dynamic routing protocol. It quickly detects topological changes in the switch fabric (such as, switch interface failures) and calculates new loop-free routes after a period of convergence. This period of convergence is short and involves a minimum of routing traffic.

All switches in the fabric run the same algorithm and maintain identical databases describing the switch fabric topology. This database contains each switch's local state, including its usable interfaces and reachable neighbors. Each switch distributes its local state throughout the switch fabric by flooding. From the topological database, each switch constructs a set of best path trees (using itself as the root) that specify routes to all other switches in the fabric.

2.1 Definitions of Commonly Used Terms

This section contains a collection of definitions for terms that have a specific meaning to the protocol and that are used throughout the text.

Switch ID

A 10-octet value that uniquely identifies the switch within the switch fabric. The value consists of the 6-octet base MAC address of the switch, followed by 4 octets of zeroes.

Network link

The physical connection between two switches. A link is associated with a switch interface.

There are two physical types of network links supported by VLSP:

- Point-to-point links that join a single pair of switches. A serial line is an example of a point-to-point network link.

- Multi-access broadcast links that support the attachment of multiple switches, along with the capability to address a single message to all the attached switches. An attached ethernet is an example of a multi-access broadcast network link.

  A single topology can contain both types of links. At startup, all links are assumed to be point-to-point. A link is determined to be multi-access when more than one neighboring switch is discovered on the link.

Interface

The port over which a switch accesses one of its links. Interfaces are identified by their interface ID, a 10-octet value consisting of the 6-octet base MAC address of the switch, followed by the 4-octet local port number of the interface.

Neighboring switches

Two switches attached to a common link.

Adjacency

A relationship formed between selected neighboring switches for the purpose of exchanging routing information. Not every pair of neighboring switches become adjacent.

Link state advertisement

Describes the local state of a switch or a link. Each link state advertisement is flooded throughout the switch fabric. The collected link state advertisements of all switches and links form the protocol's topological database.

Designated switch

Each multi-access network link has a designated switch. The designated switch generates a link state advertisement for the link and has other special responsibilities in the running of the protocol.

The use of a designated switch permits a reduction in the number of adjacencies required on multi-access links. This in turn reduces the amount of routing protocol traffic and the size of the topological database.

The designated switch is selected during the discovery process. A designated switch is not selected for a point-to-point network link.

Backup designated switch

Each multi-access network link has a backup designated switch. The backup designated switch maintains adjacencies with the same switches on the link as the designated switch. This optimizes the failover time when the backup designated switch must take over for the (failed) designated switch.

The backup designated switch is selected during the Discovery process. A backup designated switch is not selected for a point-to-point network link.

2.2 Differences Between VLSP and OSPF

The VLS protocol is derived from the OSPF link-state routing protocol described in [RFC2328].

2.2.1 Operation at the Physical Layer

The primary differences between the VLS and OSPF protocols stem from the fact that OSPF runs over the IP layer, while VLSP runs at the physical MAC layer. This difference has the following repercussions:

- VLSP does not support features (such as fragmentation) that are typically provided by network layer service providers.

- Due to the unrelated nature of MAC address assignments, VLSP provides no summarization of the address space (such as, classical IP subnet information) or level 2 routing (such as,

  IS-IS Phase V DECnet). Thus, VLSP does not support grouping switches into areas. All switches exist in a single area. Since a single domain exists within any switch fabric, there is no need for VLSP to provide interdomain reachability.

- As mentioned in Section 10.1.1, ISMP uses a single well-known multicast address for all packets. However, parts of the VLS protocol (as derived from OSPF) are dependent on certain network layer addresses -- in particular, the AllSPFSwitches and AllDSwitches multicast addresses that drive the distribution of link state advertisements throughout the switch fabric. In order to facilitate the implementation of the protocol at the physical MAC layer, network layer address information is encapsulated in the protocol packets (see Section 10.3). This information is unbundled and packets are then processed as if they had been sent or received on that multicast address.

2.2.2 All Links Treated as Point-to-Point

When the switch first comes on line, VLSP assumes all network links are point-to-point and no more than one neighboring switch will be discovered on any one port. Therefore, at startup, VLSP does not send its own Hello packets over its network ports, but instead, relies on the VlanHello protocol [IDhello] for the discovery of its neighbor switches. If a second neighbor is detected on a link, the link is then deemed multi-access and the interface type is changed to broadcast. At that point, VLSP exchanges its own Hello packets with the switches on the link in order to select a designated switch and designated backup switch for the link.

This method eliminates unnecessary duplication of message traffic and processing, thereby increasing the overall efficiency of the switch fabric.

Note: Previous versions of VLSP treated all links as if they were broadcast (multi-access). Thus, if VLSP determines that a neighbor switch is running an older version of the protocol software (see Section 6.1), it will change the interface type to broadcast and begin exchanging Hello packets with the single neighbor switch.

2.2.3 Routing Path Information

Instead of providing the next hop to a destination, VLSP calculates and maintains complete end-to-end path information. On request, a list of individual port identifiers is generated describing a complete path from the source switch to the destination switch. If multiple equal-cost routes exist to a destination switch, up to three paths are calculated and returned.

2.2.4 Configurable Parameters

OSPF supports (and requires) configurable parameters. In fact, even the default OSPF configuration requires that IP address assignments be specified. On the other hand, no configuration information is ever required for the VLS protocol. Switches are uniquely identified by their base MAC addresses and ports are uniquely identified by the base MAC address of the switch and a port number.

While a developer is free to implement configurable parameters for the VLS protocol, the current version of VLSP supports configurable path metrics only. Note that this has the following repercussions:

- All switches are assigned a switch priority of 1. This forces the selection of the designated switch to be based solely on base MAC address.

- Authentication is not supported.

2.2.5 Features Not supported

In addition to those features mentioned in the previous sections, the following OSPF features are not supported by the current version of VLSP:

- Periodic refresh of link state advertisements. (This optimizes performance by eliminating unnecessary traffic between the switches.)

- Routing based on non-zero type of service (TOS).

- Use of external routing information for destinations outside the switch fabric.

2.3 Functional Summary

There are essentially four operational stages of the VLS protocol.

- Discovery Process The discovery process involves two steps:
  - Neighboring switches are detected by the VlanHello protocol [IDhello] which then notifies VLSP of the neighbor.
  - If more than one neighbor switch is detected on a single port, the link is determined to be multi-access. VLSP then sends its own Hello packets over the link in order to discover the full set of neighbors on the link and select a designated switch and designated backup switch for the link. Note that this selection process is unnecessary on point-to-point links.

  The discovery process is described in more detail in Section 6.

- Synchronizing the Databases

  Adjacencies are used to simplify and speed up the process of synchronizing the topological database (also known as the link state database) maintained by each switch in the fabric. Each switch is only required to synchronize its database with those neighbors to which it is adjacent. This reduces the amount of routing protocol traffic across the fabric, particularly for multi-access links with multiple switches.

  The process of synchronizing the databases is described in more detail in Section 7.

- Maintaining the Databases

  Each switch advertises its state (also known as its link state) any time its link state changes. Link state advertisements are distributed throughout the switch fabric using a reliable flooding algorithm that ensures that all switches in the fabric are notified of any link state changes.

  The process of maintaining the databases is described in more detail in Section 8.

o Calculating the Best Paths

The link state database consists of the collection of link state advertisements received from each switch. Each switch uses its link state database to calculate a set of best paths, using itself as root, to all other switches in the fabric.

The process of recalculating the set of best paths is described in more detail in Section 9.

2.4 Protocol Packets

In addition to the frame header and the ISMP packet header described in Section 10.1, all VLS protocol packets share a common protocol header, described in Section 10.4.

The VLSP packet types are listed below in Table 1. Their formats are described in Section 10.6.

Type	Packet Name	Protocol Function
1	Hello	Select DS and Backup DS
2	Database Description	Summarize database contents
3	Link State Request	Database download
4	Link State Update	Database update
5	Link State Ack	Flooding acknowledgment

Table 1: VLSP Packet Types

The Hello packets are used to select the designated switch and the backup designated switch on multi-access links. The Database Description and Link State Request packets are used to form adjacencies. Link State Update and Link State Acknowledgment packets are used to update the topological database.

Each Link State Update packet carries a set of link state advertisements. A single Link State Update packet may contain the link state advertisements of several switches. There are two different types of link state advertisement, as shown below in Table 2.

LS Type	Advertisement Name	Advertisement Description
1	Switch link advertisements	Originated by all switches. This advertisement describes the collected states of the switch's interfaces.
2	Network link advertisements	Originated by the designated switch. This advertisement contains the list of switches connected to the network link.

Table 2: VLSP Link State Advertisements

2.5 Protocol Data Structures

The VLS protocol is described in this specification in terms of its operation on various protocol data structures. Table 3 lists the primary VLSP data structures, along with the section in which they are described in detail.

Structure Name	Description
Interface Data Structure	Section 3
Neighbor Data Structure	Section 4
Area Data Structure	Section 5

Table 3: VLSP Data Structures

2.6 Basic Implementation Requirements

An implementation of the VLS protocol requires the following pieces of system support:

Timers

Two types of timer are required. The first type, known as a one-shot timer, expires once and triggers an event. The second type, known as an interval timer, expires at preset intervals. Interval timers are used to trigger events at periodic intervals. The granularity of both types of timers is one second.

Interval timers should be implemented in such a way as to avoid drift. In some switch implementations, packet processing can affect timer execution. For example, on a multi-access link with multiple switches, regular broadcasts can lead to undesirable synchronization of routing packets unless the interval timers have been implemented to avoid drift. If it is not possible to

implement drift-free timers, small random amounts of time should be added to or subtracted from the timer interval at each firing.

List manipulation primitives

Much of the functionality of VLSP is described here in terms of its operation on lists of link state advertisements. Any particular advertisement may be on many such lists. Implementation of VLSP must be able to manipulate these lists, adding and deleting constituent advertisements as necessary.

Tasking support

Certain procedures described in this specification invoke other procedures. At times, these other procedures should be executed in-line -- that is, before the current procedure has finished. This is indicated in the text by instructions to "execute" a procedure. At other times, the other procedures are to be executed only when the current procedure has finished. This is indicated by instructions to "schedule" a task. Implementation of VLSP must provide these two types of tasking support.

2.7 Organization of the Remainder of This Document

The remainder of this document is organized as follows:

- Section 3 through Section 5 describe the primary data structures used by the protocol. Note that this specification is presented in terms of these data structures in order to make explanations more precise. Implementations of the protocol must support the functionality described, but need not use the exact data structures that appear in this specification.

- Section 6 through Section 9 describe the four operational stages of the protocol: the discovery process, synchronizing the databases, maintaining the databases, and calculating the set of best paths.

- Section 10 describes the processing of VLSP packets and presents detailed descriptions of their formats.

- Section 11 presents detailed descriptions of link state advertisements.

- Section 12 summarizes the protocol parameters.

3. Interface Data Structure

The port over which a switch accesses a network link is known as the link interface. Each switch maintains a separate interface data structure for each network link.

The following data items are associated with each interface:

Type

The type of network to which the interface is attached -- point-to-point or broadcast (multi-access). This data item is initialized to point-to-point when the interface becomes operational. If a second neighbor is detected on the link after the first neighbor has been discovered, the link interface type is changed to broadcast. The type remains as broadcast until the interface is declared down, at which time the type reverts to point-to-point.

Note: Previous versions of VLSP treated all links as if they were multi-access. Thus, if VLSP determines that a neighbor switch is running an older version of the protocol software (see Section 6.1), it will change the interface type to broadcast.

State

The functional level of the interface. The state of the interface is included in all switch link advertisements generated by the switch, and is also used to determine whether full adjacencies are allowed on the interface. See Section 3.1 for a complete description of interface states.

Interface identifier

A 10-octet value that uniquely identifies the interface. This value consists of the 6-octet base MAC address of the neighbor switch, followed by the 4-octet local port number of the interface.

Area ID

A 4-octet value identifying the area. Since VLSP does not support multiple areas, the value here is always zero.

HelloInterval

The interval, in seconds, at which the switch sends VLSP Hello packets over the interface. This parameter is not used on point-to-point links.

SwitchDeadInterval

The length of time, in seconds, that neighboring switches will wait before declaring the local switch dNeighboring switches

A list of the neighboring switches attached to this network link. This list is created during the discovery process. Adjacencies are formed to one or more of these neighbors. The set of adjacent neighbors can be determined by examining the states of the neighboring switches as shown in their link state advertisements.

Designated switch

The designated switch selected for the multi-access network link. (A designated switch is not selected for a point-to-point link.) This data item is initialized to zero when the switch comes on-line, indicating that no designated switch has been chosen for the link.

Backup designated switch

The backup designated switch selected for the multi-access network link. (A backup designated switch is not selected for a point-to-point link.) This data item is initialized to zero when the switch comes on-line, indicating that no backup designated switch has been chosen for the link.

Interface output cost(s)

The cost of sending a packet over the interface. The link cost is expressed in the link state metric and must be greater than zero.

RxmtInterval

The number of seconds between link state advertisement retransmissions, for adjacencies belonging to this interface. This value is also used to time the retransmission of Database Description and Link State Request packets.

3.1 Interface States

This section describes the various states of a switch interface. The states are listed in order of progressing functionality. For example, the inoperative state is listed first, followed by a list of the intermediate states through which the interface passes before attaining the final, fully functional state. The specification makes use of this ordering by references such as "those interfaces in state greater than X".

Figure 1 represents the interface state machine, showing the progression of interface state changes. The arrows on the graph represent the events causing each state change. These events are described in Section 3.2. The interface state machine is described in detail in Section 3.3.

Down

This is the initial state of the interface. In this state, the interface is unusable, and no protocol traffic is sent or received on the interface. In this state, interface parameters are set to their initial values, all interface timers are disabled, and no adjacencies are associated with the interface.

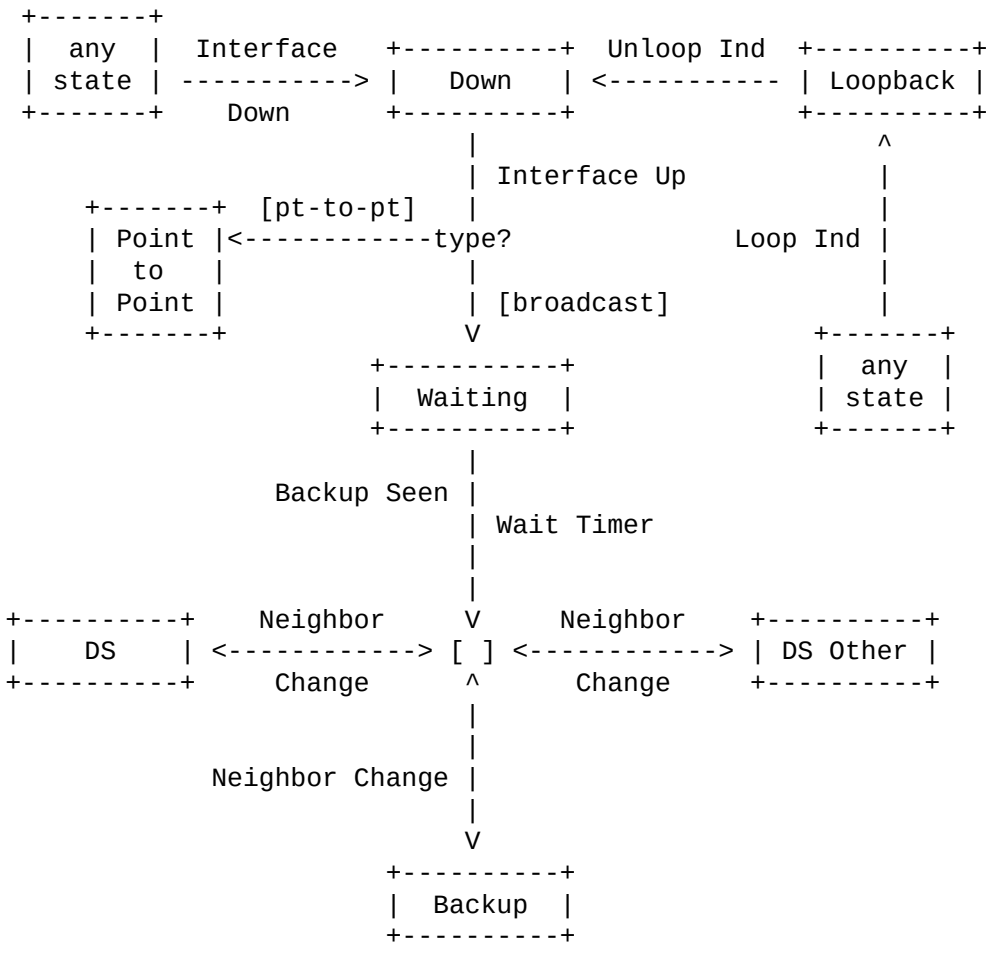


Figure 1: Interface State Machine

Loopback

In this state, the switch interface is looped back, either in hardware or in software. The interface is unavailable for regular data traffic.

Point-to-Point

In this state, the interface is operational and is connected to a physical point-to-point link. On entering this state, the switch attempts to form an adjacency with the neighboring switch.

Waiting

In this state, the switch is attempting to identify the backup designated switch for the link by monitoring the Hello packets it receives. The switch does not attempt to select a designated switch or a backup designated switch until it changes out of this state, thereby preventing unnecessary changes of the designated switch and its backup.

DS Other

In this state, the interface is operational and is connected to a multi-access broadcast link on which other switches have been selected as the designated switch and the backup designated switch. On entering this state, the switch attempts to form adjacencies with both the designated switch and the backup designated switch.

Backup

In this state, the switch itself is the backup designated switch on the attached multi-access broadcast link. It will be promoted to designated switch if the current designated switch fails. The switch establishes adjacencies with all other switches attached to the link. (See Section 6.3 for more information on the functions performed by the backup designated switch.)

DS

In this state, this switch itself is the designated switch on the attached multi-access broadcast link. The switch establishes adjacencies with all other switches attached to the link. The switch is responsible for originating network link advertisements for the link, containing link information for all switches attached to the link, including the designated switch itself. (See Section 6.3 for more information on the functions performed by the designated switch.)

3.2 Events Causing Interface State Changes

The state of an interface changes due to an interface event. This section describes these events.

Interface events are shown as arrows in Figure 1, the graphic representation of the interface state machine. For more information on the interface state machine, see Section 3.3.

Interface Up

This event is generated by the VlanHello protocol [IDhello] when it discovers a neighbor switch on the interface. The interface is now operational. This event causes the interface to change out of the Down state. The state it enters is determined by the interface type. If the interface type is broadcast (multi-access), this event also causes the switch to begin sending periodic Hello packets out over the interface.

Wait Timer

This event is generated when the one-shot Wait timer expires, triggering the end of the required waiting period before the switch can begin the process of selecting a designated switch and a backup designated switch on a multi-access link.

Backup Seen

This event is generated when the switch has detected the existence or non-existence of a backup designated switch for the link, as determined in one of the following two ways:

- A Hello packet has been received from a neighbor that claims to be the backup designated switch.

- A Hello packet has been received from a neighbor that claims to be the designated switch. In addition, the packet indicated that there is no backup.

In either case, the interface must have bidirectional communication with its neighbor -- that is, the local switch must be listed in the neighbor's Hello packet.

This event signals the end of the Waiting state.

Neighbor change

This event is generated when there has been one of the following changes in the set of bidirectional neighbors associated with the interface. (See Section 4.1 for information on neighbor states.)

- Bidirectional communication has been established with a neighbor -- the state of the neighbor has changed to 2-Way or higher.

- Bidirectional communication with a neighbor has been lost -- the state of the neighbor has changed to Init or lower.

- A bidirectional neighbor has just declared itself to be either the designated switch or the backup designated switch, as detected by examination of that neighbor's Hello packets.

- A bidirectional neighbor is no longer declaring itself to be either the designated switch or the backup designated switch, as detected by examination of that neighbor's Hello packets.

- The advertised switch priority of a bidirectional neighbor has changed, as detected by examination of that neighbor's Hello packets.

When this event occurs, the designated switch and the backup designated switch must be reselected.

Loop Ind

This event is generated when an interface enters the Loopback state. This event can be generated by either the network management service or by the lower-level protocols.

Unloop Ind

This event is generated when an interface leaves the Loopback state. This event can be generated by either the network management service or by the lower-level protocols.

Interface Down

This event is generated under the following two circumstances:

- The VlanHello [IDhello] protocol has determined that the interface is no longer functional.

- The neighbor state machine has detected a second neighboring switch on a link presumed to be of type point-to-point. In addition to generating the Interface Down event, the neighbor state machine changes the interface type to broadcast.

In both instances, this event forces the interface state to Down. However, when the event is generated by the neighbor state machine, it is immediately followed by an Interface Up event. (See Section 4.3.)

3.3 Interface State Machine

This section presents a detailed description of the interface state machine.

Interface states (see Section 3.1) change as the result of various events (see Section 3.2). However, the effect of each event can vary, depending on the current state of the interface. For this reason, the state machine described in this section is organized according to the current interface state and the occurring event. For each state/event pair, the new interface state is listed, along with a description of the required processing.

Note that when the state of an interface changes, it may be necessary to originate a new switch link advertisement. See Section 8.1 for more information.

Some of the processing described here includes generating events for the neighbor state machine. For example, when an interface becomes inoperative, all neighbor connections associated with the interface must be destroyed. For more information on the neighbor state machine, see Section 4.3.

State(s): Down
Event: Interface Up
New state: Depends on action routine
Action:
   If the interface is a point-to-point link, set the interface state to Point-to-Point. Otherwise, start the Hello interval timer, enabling the periodic sending of Hello packets over the interface. If the switch is not eligible to become the designated switch, change the interface state to DS Other. Otherwise, set the interface state to Waiting and start the one-shot wait timer. Create a new neighbor data structure for the neighbor switch, initialize all neighbor parameters and set the stateof the neighbor to Down.

State(s): Waiting
Event: Backup Seen
New state: Depends on action routine
Action:
   Select the designated switch and backup designated switch for the attached link, as described in Section 6.3.1. As a result of this selection, set the new state of the interface to either DS Other, Backup or DS.

State(s): Waiting
Event: Wait Timer
New state: Depends on action routine
Action:
Select the designated switch and backup designated switch for the attached link, as described in Section 6.3.1. As a result of this selection, set the new state of the interface to either DS Other, Backup or DS.

State(s): DS Other, Backup or DS
Event: Neighbor Change
New state: Depends on action routine
Action:
Reselect the designated switch and backup designated switch for the attached link, as described in Section 6.3.1. As a result of this selection, set the new state of the interface to either DS Other, Backup or DS.

State(s): Any State
Event: Interface Down
New state: Down
Action:
Reset all variables in the interface data structure and disable all timers. In addition, destroy all neighbor connections associated with the interface by generating the KillNbr event on all neighbors listed in the interface data structure.

State(s): Any State
Event: Loop Ind
New state: Loopback
Action:
Reset all variables in the interface data structure and disable all timers. In addition, destroy all neighbor connections associated with the interface by generating the KillNbr event on all neighbors listed in the interface data structure.

State(s): Loopback
Event: Unloop Ind
New state: Down
Action:
No action is necessary beyond changing the interface state to Down because the interface was reset on entering the Loopback state.

4. Neighbor Data Structure

Each switch conducts a conversation with its neighboring switches and each conversation is described by a neighbor data structure. A conversation is associated with a switch interface, and is identified by the neighboring switch ID.

Note that if two switches have multiple attached links in common, multiple conversations ensue, each described by a unique neighbor data structure. Each separate conversation is treated as a separate neighbor.

The neighbor data structure contains all information relevant to any adjacency formed between the two neighbors. Remember, however, that not all neighbors become adjacent. An adjacency can be thought of as a highly developed conversation between two switches.

State

The functional level of the neighbor conversation. See Section 4.1 for a complete description of neighbor states.

Inactivity timer

A one-shot timer used to determine when to declare the neighbor down if no Hello packet is received from this (multi-access) neighbor. The length of the timer is SwitchDeadInterval seconds, as contained in the neighbor's Hello packet. This timer is not used on point-to-point links.

Master/slave flag

A flag indicating whether the local switch is to act as the master or the slave in the database exchange process (see Section 7.2). The master/slave relationship is negotiated when the conversation changes to the ExStart state.

Sequence number

A 4-octet number identifying individual Database Description packets. When the neighbor state ExStart is entered and the database exchange process is started, the sequence number is set to a value not previously seen by the neighboring switch. (One possible scheme is to use the switch's time of day counter.) The sequence number is then incremented by the master with each new Database Description packet sent. See Section 7.2 for more information on the database exchange process.

Neighbor ID

The switch ID of the neighboring switch, as discovered by the VlanHello protocol [IDhello] or contained in the neighbor's Hello packets.

Neighbor priority

The switch priority of the neighboring switch, as contained in the neighbor's Hello packets. Switch priorities are used when selecting the designated switch for the attached multi-access link. Priority is not used on point-to-point links.

Interface identifier

A 10-octet value that uniquely identifies the interface over which this conversation is being held. This value consists of the 6-octet base MAC address of the neighbor switch, followed by the 4-octet local port number of the interface.

Neighbor's designated switch

The switch ID identifying the neighbor's idea of the designated switch, as contained in the neighbor's Hello packets. This value is used in the local selection of the designated switch. It is not used on point-to-point links.

Neighbor's backup designated switch

The switch ID identifying the neighbor's idea of the backup designated switch, as contained in the neighbor's Hello packets. This value is used in the local selection of the backup designated switch. It is not used on point-to-point links.

Link state retransmission list

The list of link state advertisements that have been forwarded over but not acknowledged on this adjacency. The local switch retransmits these link state advertisements at periodic intervals until they are acknowledged or until the adjacency is destroyed. (For more information on retransmitting link state advertisements, see Section 8.2.5.)

Database summary list

The set of link state advertisement headers that summarize the local link state database. When the conversation changes to the Exchange state, this list is sent to the neighbor via Database Description packets. (For more information on the synchronization of databases, see Section 7.)

Link state request list

The list of link state advertisements that must be received in order to synchronize with the neighbor switch's link state database. This list is created as Database Description packets are received, and is then sent to the neighbor in Link State Request packets. (For more information on the synchronization of databases, see Section 7.)

4.1 Neighbor States

This section describes the various states of a conversation with a neighbor switch. The states are listed in order of progressing functionality. For example, the inoperative state is listed first, followed by a list of the intermediate states through which the conversation passes before attaining the final, fully functional state. The specification makes use of this ordering by references such as "those neighbors/adjacencies in state greater than X".

Figure 2 represents the neighbor state machine. The arrows on the graph represent the events causing each state change. These events are described in Section 4.2. The neighbor state machine is described in detail in Section 4.3.

Down

This is the initial state of a neighbor conversation.

Init

In this state, the neighbor has been discovered, but bidirectional communication has not yet been established. All neighbors in this state or higher are listed in the VLS Hello packets sent by the local switch over the associated (multi-access) interface.

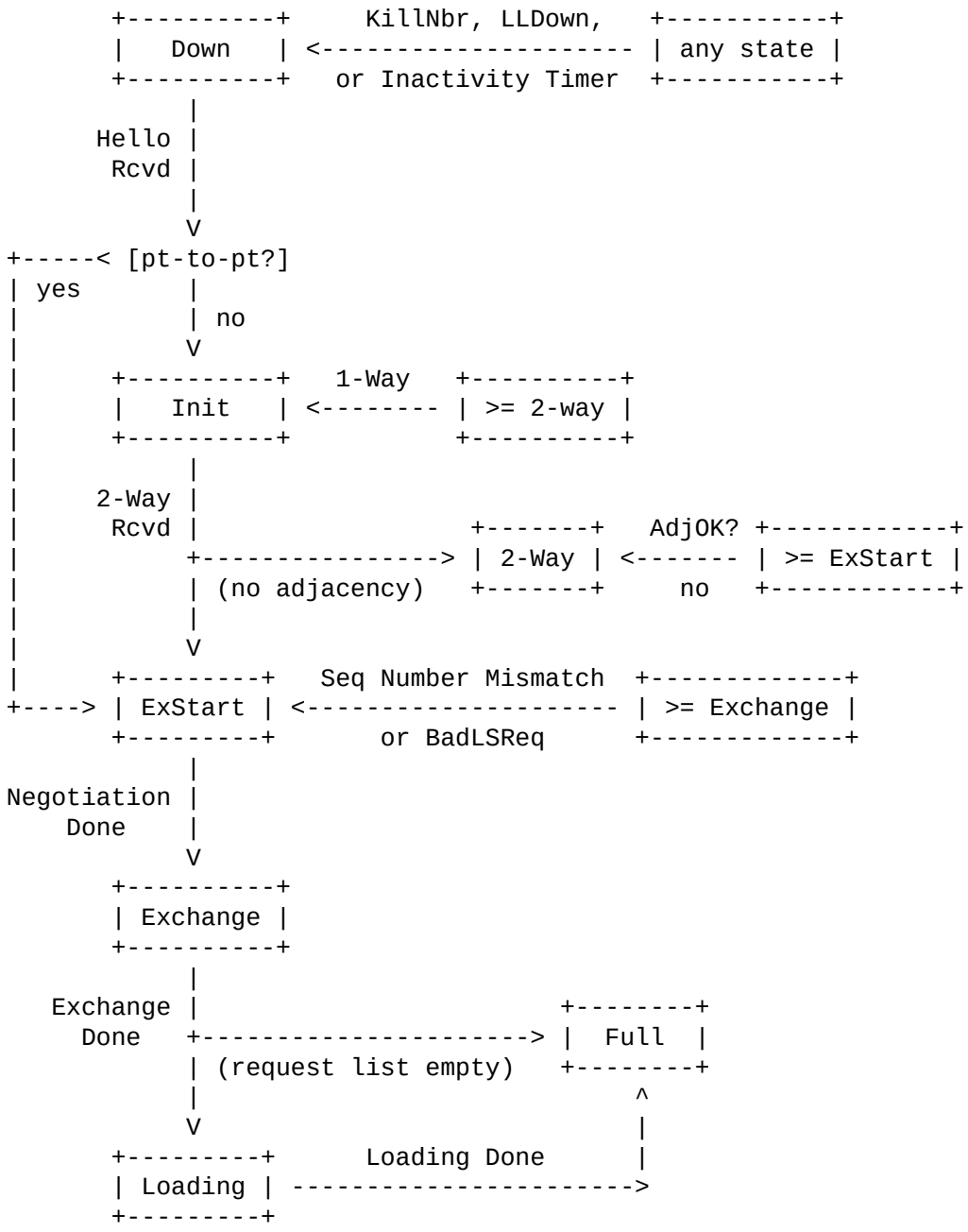


Figure 2: Neighbor State Machine

2-Way

In this state, communication between the two switches is bidirectional. This is the most advanced state short of beginning to establish an adjacency. On a multi-access link, the designated switch and the backup designated switch are selected from the set of neighbors in state 2-Way or greater.

ExStart

This state indicates that the two switches have begun to establish an adjacency by determining which switch is the master, as well as the initial sequence number for Database Descriptor packets. Neighbor conversations in this state or greater are called adjacencies.

Exchange

In this state, the switches are exchanging Database Description packets. (See Section 7.2 for a complete description of this process.) All adjacencies in the Exchange state or greater are used by the distribution procedure (see Section 8.2), and are capable of transmitting and receiving all types of VLSP routing packets.

Loading

In this state, the local switch is sending Link State Request packets to the neighbor asking for the more recent advertisements that were discovered in the Exchange state.

Full

In this state, the two switches are fully adjacent. These adjacencies will now appear in switch link and network link advertisements generated for the link.

4.2 Events Causing Neighbor State Changes

The state of a neighbor conversation changes due to neighbor events. This section describes these events.

Neighbor events are shown as arrows in Figure 2, the graphic representation of the neighbor state machine. For more information on the neighbor state machine, see Section 4.3.

Hello Received

This event is generated when a Hello packet has been received from a neighbor.

2-Way Received

This event is generated when the local switch sees its own switch ID listed in the neighbor's Hello packet, indicating that bidirectional communication has been established between the two switches.

Negotiation Done

This event is generated when the master/slave relationship has been successfully negotiated and initial packet sequence numbers have been exchanged. This event signals the start of the database exchange process (see Section 7.2).

Exchange Done

This event is generated when the database exchange process is complete and both switches have successfully transmitted a full sequence of Database Description packets. (For more information on the database exchange process, see Section 7.2.)

BadLSReq

This event is generated when a Link State Request has been received for a link state advertisement that is not contained in the database. This event indicates an error in the synchronization process.

Loading Done

This event is generated when all Link State Updates have been received for all out-of-date portions of the database. (See Section 7.3.)

AdjOK?

This event is generated when a decision must be made as to whether an adjacency will be established or maintained with the neighbor. This event will initiate some adjacencies and destroy others.

Seq Number Mismatch

This event is generated when a Database Description packet has been received with any of the following conditions:

- The packet contains an unexpected sequence number.
- The packet (unexpectedly) has the Init bit set.
- The packet has a different Options field than was previously seen.

These conditions all indicate that an error has occurred during the establishment of the adjacency.

1-Way

This event is generated when bidirectional communication with the neighbor has been lost. That is, a Hello packet has been received from the neighbor in which the local switch is not listed.

KillNbr

This event is generated when further communication with the neighbor is impossible.

Inactivity Timer

This event is generated when the inactivity timer has expired, indicating that no Hello packets have been received from the neighbor in SwitchDeadInterval seconds. This timer is used only on broadcast (multi-access) links.

LLDown

This event is generated by the lower-level switch discovery protocols and indicates that the neighbor is now unreachable.

4.3 Neighbor State Machine

This section presents a detailed description of the neighbor state machine.

Neighbor states (see Section 4.1) change as the result of various events (see Section 4.2). However, the effect of each event can vary, depending on the current state of the conversation with the neighbor. For this reason, the state machine described in this section is organized according to the current neighbor state and the occurring event. For each state/event pair, the new neighbor state is listed, along with a description of the required processing.

Note that when the neighbor state changes as a result of an interface Neighbor Change event (see Section 3.2), it may be necessary to rerun the designated switch selection algorithm. In addition, if the interface associated with the neighbor conversation is in the DS state (that is, the local switch is the designated switch), changes in the neighbor state may cause a new network link advertisement to be originated (see Section 8.1).

When the neighbor state machine must invoke the interface state machine, it is invoked as a scheduled task. This simplifies processing, by ensuring that neither state machine executes recursively.

State(s): Down
Event: Hello Received
New state: Depends on the interface type
Action:
If the interface type of the associated link is point-to-point, change the neighbor state to ExStart. Otherwise, change the neighbor state to Init and start the inactivity timer for the neighbor. If the timer expires before another Hello packet is received, the neighbor switch is declared dead.

State(s): Init or greater
Event: Hello Received
New state: No state change
Action:
If the interface type of the associated link is point-to-point, determine whether this notification is for a different neighbor than the one previously seen. If so, generate an Interface Down event for the associated interface, reset the interface type to broadcast and generate an Interface Up event.

Note: This procedure of generating an Interface Down event and changing the interface type to broadcast is also executed if the neighbor for whom the notification was received is running an older version of the protocol software (see Section 6.1). In previous versions of the protocol, all interfaces were treated as if they were broadcast.

If the interface type is broadcast, reset the inactivity timer for the neighbor.

State(s): Init
Event: 2-Way Received
New state: Depends on action routine
Action:
Determine whether an adjacency will be formed with the neighbor (see Section 6.4). If no adjacency is to be formed, change the neighbor state to 2-Way.

Otherwise, change the neighbor state to ExStart. Initialize the sequence number for this neighbor and declare the local switch to be master for the database exchange process. (See Section 7.2.)

State(s): ExStart
Event: Negotiation Done
New state: Exchange
Action:
The Negotiation Done event signals the start of the database exchange process. See Section 7.2 for a detailed description of this process.

State(s): Exchange
Event: Exchange Done
New state: Depends on action routine
Action:
If the neighbor Link state request list is empty, change the neighbor state to Full. This is the adjacency's final state.

Otherwise, change the neighbor state to Loading. Begin sending Link State Request packets to the neighbor requesting the most recent link state advertisements, as discovered during the database exchange process. (See Section 7.2.) These advertisements are listed in the link state request list associated with the neighbor.

State(s): Loading
Event: Loading Done
New state: Full
Action:
No action is required beyond changing the neighbor state to Full. This is the adjacency's final state.

State(s): 2-Way
Event: AdjOK?
New state: Depends on action routine
Action:
If no adjacency is to be formed with the neighboring switch (see Section 6.4), retain the neighbor state at 2-Way. Otherwise, change the neighbor state to ExStart. Initialize the sequence number for this neighbor and declare the local switch to be master for the database exchange process. (See Section 7.2.)

State(s): ExStart or greater
Event: AdjOK?
New state: Depends on action routine
Action:
If an adjacency should still be formed with the neighboring switch (see Section 6.4), no state change and no further action is necessary. Otherwise, tear down the (possibly partially formed) adjacency. Clear the link state retransmission list, database summary list and link state request list and change the neighbor state to 2-Way.

State(s): Exchange or greater
Event: Seq Number Mismatch
New state: ExStart
Action:
Tear down the (possibly partially formed) adjacency. Clear the link state retransmission list, database summary list and link state request list. Change the neighbor state to ExStart and make another attempt to establish the adjacency.

State(s): Exchange or greater
Event: BadLSReq
New state: ExStart
Action:
Tear down the (possibly partially formed) adjacency. Clear the link state retransmission list, database summary list and link state request list. Change the neighbor state to ExStart and make another attempt to establish the adjacency.

State(s): Any state
Event: KillNbr
New state: Down
Action:
Terminate the neighbor conversation. Disable the inactivity timer and clear the link state retransmission list, database summary list and link state request list.

State(s): Any state
Event: LLDown
New state: Down
Action:
Terminate the neighbor conversation. Disable the inactivity timer and clear the link state retransmission list, database summary list and link state request list.

State(s): Any state
Event: Inactivity Timer
New state: Down
Action:
Terminate the neighbor conversation. Disable the inactivity timer and clear the link state retransmission list, database summary list and link state request list.

State(s): 2-Way or greater
Event: 1-Way Received
New state: Init
Action:
Tear down the adjacency between the switches, if any. Clear the link state retransmission list, database summary list and link state request list.

State(s): 2-Way or greater
Event: 2-Way received
New state: No state change
Action:
No action required.

State(s): Init
Event: 1-Way received
New state: No state change
Action:
No action required.

5. Area Data Structure

The area data structure contains all the information needed to run the basic routing algorithm. One of its components is the link state database -- the collection of all switch link and network link advertisements generated by the switches.

The area data structure contains the following items:

Area ID

A 4-octet value identifying the area. Since VLSP does not support multiple areas, the value here is always zero.

Associated switch interfaces

A list of interface IDs of the local switch interfaces connected to network links.

Link state database

The collection of all current link state advertisements for the switch fabric. This collection consists of the following:

Switch link advertisements

A list of the switch link advertisements for all switches in the fabric. Switch link advertisements describe the state of each switch's interfaces.

Network link advertisements

A list of the network link advertisements for all multi-access network links in the switch fabric. Network link advertisements describe the set of switches currently connected to each link.

Best path(s)

A set of end-to-end hop descriptions for all equal-cost best paths from the local switch to every other switch in the fabric. Each hop is specified by the interface ID of the next link in the path. Best paths are derived from the collected switch link and network link advertisements using the Dijkstra algorithm. [Perlman]

5.1 Adding and Deleting Link State Advertisements

The link state database within the area data structure must contain, at most, a single instance of each link state advertisement. To keep the database current, a switch adds link state advertisements to the database under the following conditions:

- When a link state advertisement is received during the distribution process
  - When the switch itself generates a link state advertisement

(See Section 8.2.4 for information on installing link state advertisements.)

Likewise, a switch deletes link state advertisements from the database under the following conditions:

- When a link state advertisement has been superseded by a newer instance during the flooding process
- When the switch generates a newer instance of one of its self-originated advertisements

Note that when an advertisement is deleted from the link state database, it must also be removed from the link state retransmission list of all neighboring switches.

5.2 Accessing Link State Advertisements

An implementation of the VLS protocol must provide access to individual link state advertisements, based on the advertisement's type, link state identifier, and advertising switch [1]. This lookup function is invoked during the link state distribution procedure and during calculation of the set of best paths. In addition, a switch can use the function to determine whether it has originated a particular link state advertisement, and if so, with what sequence number.

5.3 Best Path Lookup

An implementation of the VLS protocol must provide access to multiple equal-cost best paths, based on the base MAC addresses of the source and destination switches. This lookup function should return up to three equal-cost paths. Paths should be returned as lists of end-to-end hop information, with each hop specified as a interface ID of the next link in the path -- the 6-octet base MAC address of the next switch and the 4-octet local port number of the link interface.

6. Discovery Process

The first operational stage of the VLS protocol is the discovery process. During this stage, each switch dynamically detects its neighboring switches and establishes a relationship with each of these neighbors. This process has the following component steps:

- Neighboring switches are detected on each functioning network interface.
- Bidirectional communication is established with each neighbor switch.
- A designated switch and backup designated switch are selected for each multi-access network link.
- An adjacent relationship is established with selected neighbors on each link.

6.1 Neighbor Discovery

When the switch first comes on line, VLSP assumes all network links are point-to-point and no more than one neighboring switch will be discovered on any one port. Therefore, at startup, VLSP relies on the VlanHello protocol [IDhello] for the discovery of its neighbor switches.

As each neighbor is detected, VlanHello triggers a Found Neighbor event, notifying VLSP that a new neighbor has been discovered. (See [IDhello] for a description of the Found Neighbor event and the information passed.) VLSP enters the neighbor switch ID in the list of known neighbors and creates a new neighbor data structure with a neighbor status of Down. A Hello Received neighbor event is then generated, which changes the neighbor state to ExStart.

There are two circumstances under which VLSP will change the type of an interface to broadcast:

- If VLSP receives a subsequent notification from VlanHello, specifying a second (different) neighbor switch on the port., the interface is then known to be multi-access. VLSP generates an Interface Down event for the interface, resets the interface type to broadcast, and then generates an Interface Up event.
- If the functional level of the neighbor switch is less than 2, the neighbor is running a previous version of the VLSP software in which all links were treated as broadcast links. VLSP immediately changes the interface type to broadcast and generates an Interface Up event.

  In both cases, VLSP assumes control of communication over the interface by exchanging its own VLSP Hello packets with the neighbors on the link.

Note: These Hello packets are in addition to the Interswitch Keepalive messages sent by VlanHello. VlanHello still continues to monitor the condition of the interface and notifies VLSP of any change.

Each Hello packet contains the following data used during the discovery process on multi-access links:

- The switch ID and priority of the sending switch
- Values specifying the interval timers to be used for sending Hello packets and deciding whether to declare a neighbor switch Down.
- The switch ID of the designated switch and the backup designated switch for the link, as understood by the sending switch
- A list of switch IDs of all neighboring switches seen so far on the link

For a detailed description of the Hello packet format, see Section 10.6.1.

When VLSP receives a Hello packet (on a broadcast link), it first attempts to identify the sending switch by matching its switch ID to one of the known neighbors listed in the interface data structure. If this is the first Hello packet received from the switch, the switch ID is entered in the list of known neighbors and a new neighbor data structure is created with a neighbor status of Down.

At this point, the remainder of the Hello packet is examined and the appropriate interface and neighbor events are generated. In all cases, a neighbor Hello Received event is generated. Other events may also be generated, triggering further steps in the discovery process or other actions, as appropriate.

For a detailed description of the interface state machine, see Section 3.3. For a detailed description of the neighbor state machine, see Section 4.3.

6.2 Bidirectional Communication

Before a conversation can proceed with a neighbor switch, bidirectional communication must be established with that neighbor. Bidirectional communication is detected in one of two ways:

- On a point-to-point link, the VlanHello protocol sees its own switch ID listed in an Interswitch Keepalive message it has received from the neighbor.

- On a multi-access link, VLSP sees its own switch ID listed in a VLSP Hello packet it has received from the neighbor.

In either case, a neighbor 2-Way Received neighbor event is generated.

Once bidirectional communication has been established with a neighbor, the local switch determines whether an adjacency will be formed with the neighbor. However, if the link is a multi-access link, a designated switch and a backup designated switch must first be selected for the link. The next section contains a description of the designated switch, the backup designated switch, and the selection process.

6.3 Designated Switch

Every multi-access network link has a designated switch. The designated switch performs the following functions for the routing protocol:

- The designated switch originates a network link advertisement on behalf of the link, listing the set of switches (including the designated switch itself) currently attached to the link. For a detailed description of network link advertisements, see Section 11.3.

- The designated switch becomes adjacent to all other switches on the link. Since the link state databases are synchronized across adjacencies, the designated switch plays a central part in the synchronization process. For a description of the synchronization process, see Section 7.

Each multi-access network link also has a backup designated switch. The primary function of the backup designated switch is to act as a standby for the designated switch. If the current designated switch fails, the backup designated switch becomes the designated switch.

To facilitate this transition, the backup designated switch forms an adjacency with every other switch on the link. Thus, when the backup designated switch must take over for the designated switch, its link state database is already synchronized with the databases of all other switches on the link.

Note: Point-to-point network links have neither a designated switch or a backup designated switch.

6.3.1 Selecting the Designated Switch

When a multi-access link interface first becomes functional, the switch sets a one-shot Wait timer (with a value of SwitchDeadInterval seconds) for the interface. The purpose of this timer is to ensure that all switches attached to the link have a chance to establish bidirectional communication before the designated switch and backup designated switch are selected for the link.

When the Wait timer is set, the interface enters the Waiting state. During this state, the switch exchanges Hello packets with its neighbors attempting to establish bidirectional communication. The interface leaves the Waiting state under one of the following conditions:

- The Wait timer expires.
- A Hello packet is received indicating that a designated switch or a backup designated switch has already been specified for the interface.

At this point, if the switch sees that a designated switch has already been selected for the link, the switch accepts that designated switch, regardless of its own switch priority and MAC address. This situation typically means the switch has come up late on a fully functioning link. Although this makes it harder to predict the identity of the designated switch on a particular link, it ensures that the designated switch does not change needlessly, necessitating a resynchronization of the databases.

If no designated switch is currently specified for the link, the switch begins the actual selection process. Note that this selection algorithm operates only on a list of neighbor switches that are eligible to become the designated switch. A neighbor is eligible to be the designated switch if it has a switch priority greater than zero and its neighbor state is 2-Way or greater. The local switch includes itself on the list of eligible switches as long as it has a switch priority greater than zero.

The selection process includes the following steps:

1. The current values of the link's designated switch and backup designated switch are saved for use in step 6.
2. The new backup designated switch is selected as follows:

a) Eliminate from consideration those switches that have declared themselves to be the designated switch.

b) If one or more of the remaining switches have declared themselves to be the backup designated switch, eliminate from consideration all other switches.

c) From the remaining list of eligible switches, select the switch having the highest switch priority as the backup designated switch. If multiple switches have the same (highest) priority, select the switch with the highest switch ID as the backup designated switch.

3. The new designated switch is selected as follows:

a) If one or more of the switches have declared themselves to be the designated switch, eliminate from consideration all other switches.

b) From the remaining list of eligible switches, select the switch having the highest switch priority as the designated switch. If multiple switches have the same (highest) priority, select the switch with the highest switch ID as the designated switch.

4. If the local switch has been newly selected as either the designated switch or the backup designated switch, or is now no longer the designated switch or the backup designated switch, repeat steps 2 and 3, above, and then proceed to step 5.

   If the local switch is now the designated switch, it will eliminate itself from consideration at step 2a when the selection of the backup designated switch is repeated. Likewise, if the local switch is now the backup designated switch, it will eliminate itself from consideration at step 3a when the selection of the designated switch is repeated. This ensures that no switch will select itself as both backup designated switch and designated switch [2].

5. Set the interface state to the appropriate value, as follows:

o If the local switch is now the designated switch, set the interface state to DS.

o If the local switch is now the backup designated switch, set the interface state to Backup.

o Otherwise, set the interface state to DS Other.

6. If either the designated switch or backup designated switch has now changed, the set of adjacencies associated with this link must be modified. Some adjacencies may need to be formed, while others may need to be broken. Generate the neighbor AdjOK? event for all neighbors with a state of 2-Way or higher to trigger a reexamination of adjacency eligibility.

Caution: If VLSP is implemented with configurable parameters, care must be exercised in specifying the switch priorities. Note that if the local switch is not itself eligible to become the designated switch (i.e., it has a switch priority of 0), it is possible that neither a backup designated switch nor a designated switch will be selected by the above procedure. Note also that if the local switch is the only attached switch that is eligible to become the designated switch, it will select itself as designated switch and there will be no backup designated switch for the link. For this reason, it is advisable to specify a default switch priority of 1 for all switches.

6.4 Adjacencies

VLSP creates adjacencies between neighboring switches for the purpose of exchanging routing information. Not every two neighboring switches will become adjacent. On a multi-access link, an adjacency is only formed between two switches if one of them is either the designated switch or the backup designated switch.

Note that an adjacency is bound to the network link that the two switches have in common. Therefore, if two switches have multiple links in common, they may also have multiple adjacencies between them.

The decision to form an adjacency occurs in two places in the neighbor state machine:

- When bidirectional communication is initially established with the neighbor.

- When the designated switch or backup designated switch on the attached link changes.

The rules for establishing an adjacency between two neighboring switches are as follows:

- On a point-to-point link, the two neighboring switches always establish an adjacency.

- On a multi-access link, an adjacency is established with the neighboring switch under one of the following conditions:

- The local switch itself is the designated switch.
- The local switch itself is the backup designated switch.
- The neighboring switch is the designated switch.
- The neighboring switch is the backup designated switch.

If no adjacency is formed between two neighboring switches, the state of the neighbor conversation remains set to 2-Way.

7. Synchronizing the Databases

In an SPF-based routing algorithm, it is important for the link state databases of all switches to stay synchronized. VLSP simplifies this process by requiring only adjacent switches to remain synchronized.

The synchronization process begins when the switches attempt to bring up the adjacency. Each switch in the adjacency describes its database by sending a sequence of Database Description packets to its neighbor. Each Database Description packet describes a set of link state advertisements belonging to the database. When the neighbor sees a link state advertisement that is more recent than its own database copy, it makes a note to request this newer advertisement.

During this exchange of Database Description packets (known as the database exchange process), the two switches form a master/slave relationship. Database Description packets sent by the master are known as polls, and each poll contains a sequence number. Polls are acknowledged by the slave by echoing the sequence number in the Database Description response packet.

When all Database Description packets have been sent and acknowledged, the database exchange process is completed. At this point, each switch in the exchange has a list of link state advertisements for which its neighbor has more recent instances. These advertisements are requested using Link State Request packets.

Once the database exchange process has completed and all Link State Requests have been satisfied, the databases are deemed synchronized and the neighbor states of the two switches are set to Full, indicating that the adjacency is fully functional. Fully functional adjacencies are advertised in the link state advertisements of the two switches [3].

7.1 Link State Advertisements

Link state advertisements form the core of the database from which a switch calculates the set of best paths to the other switches in the fabric.

Each link state advertisement begins with a standard header. This header contains three data items that uniquely identify the link state advertisement.

- The link state type. Possible values are as follows:

  1 Switch link advertisement -- describes the collected states of the switch's interfaces.

  2 Network link advertisement -- describes the set of switches attached to the network link.

- The link state ID, defined as follows:

  - For a switch link advertisement -- the switch ID of the originating switch

  - For a network link advertisement -- the switch ID of the designated switch for the link

- The switch ID of the advertising switch -- the switch that generated the advertisement

The link state advertisement header also contains three data items that are used to determine which instance of a particular link state advertisement is the most current. (See Section 7.1.1 for a description of how to determine which instance of a link state advertisement is the most current.)

- The link state sequence number

- The link state age, stored in seconds

- The link state checksum, a 16-bit unsigned value calculated for the entire contents of the link state advertisement, with the exception of the age field

The remainder of each link state advertisement contains data specific to the type of the advertisement. See Section 11 for a detailed description of the link state header, as well as the format of a switch link or network link advertisement.

7.1.1 Determining Which Link State Advertisement Is Newer

At various times while synchronizing or updating the link state database, a switch must determine which instance of a particular link state advertisement is the most current. This decision is made as follows:

- The advertisement having the greater sequence number is the most current.
- If both instances have the same sequence number, then:
  - If the two instances have different checksum values, then the instance having the larger checksum is considered the most current [4].
- If both instances have the same sequence number and the same checksum value, then:
  - If one (and only one) of the instances is of age MaxAge, then the instance of age MaxAge is considered the most current [5].
  - Else, if the ages of the two instances differ by more than MaxAgeDiff, the instance having the smaller (younger) age is considered the most current [6].
  - Else, the two instances are considered identical.

7.2 Database Exchange Process

There are two stages to the database exchange process:

- Negotiating the master/slave relationship
- Exchanging database summary information

7.2.1 Database Description Packets

Database Description packets are used to describe a switch's link state database during the database exchange process. Each Database Description packet contains a list of headers of the link state advertisements currently stored in the sending switch's database. (See Section 11.1 for a description of a link state advertisement header.)

In addition to the link state headers, each Database Description packet contains the following data items:

- A flag (the M-bit) indicating whether or not more packets are to follow. Depending on the size of the local database and the maximum size of the packet, the list of headers in any particular Database Description packet may be only a partial list of the total database. When the M-bit is set, the list of headers is only a partial list and more headers are to follow in subsequent packets.

- A flag (the I-bit) indicating whether or not this is the first Database Description packet sent for this execution of the database exchange process.

- A flag (the MS-bit) indicating whether the sending switch thinks it is the master or the slave in the database exchange process. If the flag is set, the switch thinks it is the master.

- A 4-octet sequence number for the packet.

While the switches are negotiating the master/slave relationship, they exchange "empty" Database Description packets. That is, packets that contain no link summary information. Instead, the flags and sequence number constitute the information required for the negotiation process.

See Section 10.6.2 for a more detailed description of a Database Description packet.

7.2.2 Negotiating the Master/Slave Relationship

Before two switches can begin the actual exchange of database information, they must decide between themselves who will be the master in the exchange process and who will be the slave. They must also agree on the starting sequence number for the Database Description packets.

Once a switch has decided to form an adjacency with a neighboring switch, it sets the neighbor state to ExStart and begins sending empty Database Description packets to its neighbor. These packets contain the starting sequence number the switch plans to use in the exchange process. Also, the I-bit and M-bit flags are set, as well as the MS-bit. Thus, each switch in the exchange begins by believing it will be the master.

Empty Database Description packets are retransmitted every RxmtInterval seconds until the neighbor responds.

When a switch receives an empty Database Description packet from its neighbor, it determines which switch will be the master by comparing the switch IDs. The switch with the highest switch ID becomes the master of the exchange. Based on this determination, the switch proceeds as follows:

- If the switch is to be the slave of the database exchange process, it acknowledges that it is the slave by sending another empty Database Description packet to the master. This packet contains the master's sequence number and has the MS-bit and the I-bit cleared.

- The switch then generates a neighbor event of Negotiation Done to change its neighbor state to Exchange and waits for the first non-empty Database Description packet from the master.

- If the switch is to be the master of the database exchange, it waits to receive an acknowledgment from its neighbor -- that is, an empty Database Description packet with the MS-bit and I-bit cleared and containing the sequence number it (the master) previously sent.

- When it receives the acknowledgment, it generates a neighbor event of Negotiation Done to change its neighbor state to Exchange and begin the actual exchange of Database Description packets.

Note that during the negotiation process, the receipt of an inconsistent packet will result in a neighbor event of Seq Number Mismatch, terminating the process. See Section 4.3 for more information.

7.2.3 Exchanging Database Description Packets

Once the neighbor state changes to Exchange, the switches begin the exchange of Database Description packets containing link state summary data. The process proceeds as follows:

1. The master sends a packet containing a list of link state headers. If the packet contains only a portion of the unexchanged database -- that is, more Database Description packets are to follow -- the packet has the M-bit set. The MS-bit is set and the I-bit is clear.

   If the slave does not acknowledge the packet within RxmtInterval seconds, the master retransmits the packet.

2. When the slave receives a packet, it first checks the sequence number to see if the packet is a duplicate. If so, it simply acknowledges the packet by clearing the MS-bit and returning the packet to the master. (Note that the slave acknowledges all Database Description packets that it receives, even those that are duplicates.)

   Otherwise, the slave processes the packet by doing the following:

   - For each link state header listed in the packet, the slave searches its own link state database to determine whether it has an instance of the advertisement.

   - If the slave does not have an instance of the link state advertisement, or if the instance it does have is older than the instance listed in the packet, it creates an entry in its link state request list in the neighbor data structure. See Section 7.1.1 for a description of how to determine which instance of a link state advertisement is the newest.

   - When the slave has examined all headers, it acknowledges the packet by turning the MS-bit off and returning the packet to the master.

3. When the master receives the first acknowledgment for a particular Database Description packet, it processes the acknowledgment as follows:

   - For each link state header listed in the packet, the master checks to see if the slave has indicated it has an instance of the link state advertisement that is newer than the instance the master has in its own database. If so, the master creates an entry in its link state request list in the neighbor data structure.

   - The master then increments the sequence number and sends another packet containing the next set of link state summary information, if any.

   Subsequent acknowledgments for the Database Description packet (those with the same sequence number) are discarded.

   When the master sends the last portion of its database summary information, it clears the M-bit in the packet to indicate that no more packets are to be sent.

4. When the slave receives a Database Description packet with the M-bit clear, it processes the packet, as described above in step 2. After it has completed processing and has acknowledged the packet to the master, it generates an Exchange Done neighbor event and its neighbor state changes to Loading.

   The database exchange process is now complete for the slave, and it begins the process of requesting those link state advertisements for which the master has more current instances (see Section 7.3).

5. When the master receives an acknowledgment for the final Database Description packet, it processes the acknowledgment as described above in step 3. Then it generates an Exchange Done neighbor event and its neighbor state changes to Loading.

   The database exchange process is now complete for the master, and it begins the process of requesting those link state advertisements for which the slave has more current instances (see Section 7.3).

Note that during this exchange, the receipt of an inconsistent packet will result in a neighbor event of Seq Number Mismatch, terminating the process. See Section 4.3 for more information.

7.3 Updating the Database

When either switch completes the database exchange process and its neighbor state changes to Loading, it has a list of link state advertisements for which the neighboring switch has a more recent instance. This list is stored in the neighbor data structure as the link state request list.

To complete the synchronization of its database with that of its neighbor, the switch must obtain the most current instances of those link state advertisements.

The switch requests these advertisements by sending its neighbor a Link State Request packet containing the description of one or more link state advertisement, as defined by the advertisement's type, link state ID, and advertising switch. (For a detailed description of the Link State Request packet, see Section 10.6.3.) The switch continues to retransmit this packet every RxmtInterval seconds until it receives a reply from the neighbor.

When the neighbor switch receives the Link State Request packet, it responds with a Link State Update packet containing its most current instance of each of the requested advertisements. (Note that the neighboring switch can be in any of the Exchange, Loading or Full neighbor states when it responds to a Link State Request packet.)

If the neighbor cannot locate a particular link state advertisement in its database, something has gone wrong with the synchronization process. The switch generates a BadLSReq neighbor event and the partially formed adjacency is torn down. See Section 4.3 for more information.

Depending on the size of the link state request list, it may take more than one Link State Request packet to obtain all the necessary advertisements. Note, however, that there must at most one Link State Request packet outstanding at any one time.

7.4 An Example

Figure 3 shows an example of an adjacency being formed between two switches -- S1 and S2 -- connected to a network link. S2 is the designated switch for the link and has a higher switch ID than S1.

The neighbor state changes that each switch goes through are listed on the sides of the figure.

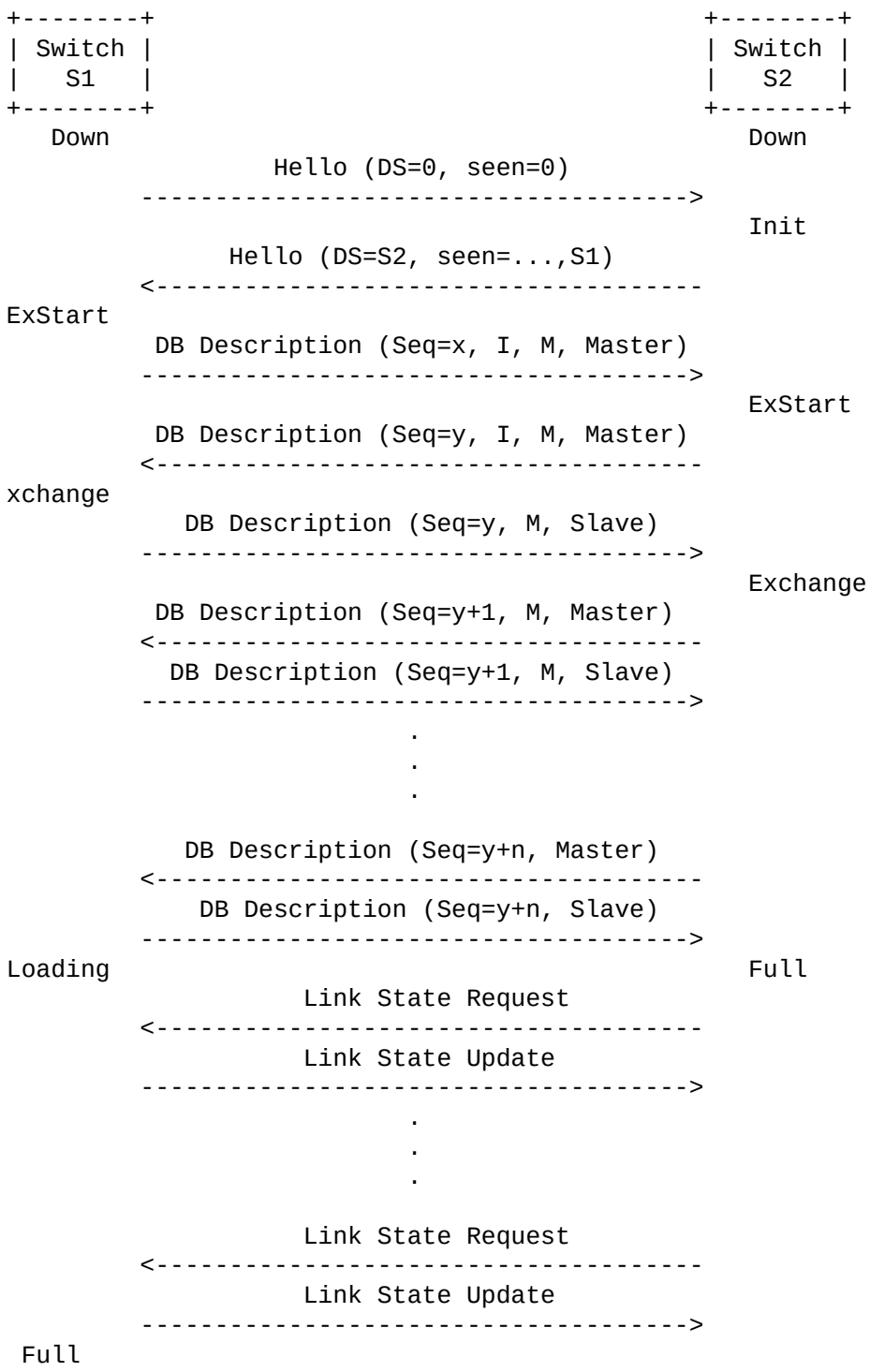


Figure 3: An Example of Bringing Up an Adjacency

At the top of Figure 3, S1's interface to the link becomes operational, and S1 begins sending Hello packets over the interface. At this point, S1 does not yet know the identity of the designated switch or of any other neighboring switches. S2 receives the Hello packet from S1 and changes its neighbor state to Init. In its next Hello packet, S2 indicates that it is itself the designated switch and that it has received a Hello packet from S1. S1 receives the Hello packet and changes its state to ExStart, starting the process of bringing up the adjacency.

S1 begins by asserting itself as the master. When it sees that S2 is indeed the master (because of S2's higher switch ID), S1 changes to slave and adopts S2's sequence number. Database Description packets are then exchanged, with polls coming from the master (S2) and acknowledgments from the slave (S1). This sequence of Database Description packets ends when both the poll and associated acknowledgment have the M-bit off.

In this example, it is assumed that S2 has a completely up-to-date database and immediately changes to the Full state. S1 will change to the Full state after updating its database by sending Link State Request packets and receiving Link State Update packets in response.

Note that in this example, S1 has waited until all Database Description packets have been received from S2 before sending any Link State Request packets. However, this need not be the case. S1 could interleave the sending of Link State Request packets with the reception of Database Description packets.

8. Maintaining the Databases

Each switch advertises its state (also known as its link state) by originating switch link advertisements. In addition, the designated switch on each network link advertises the state of the link by originating network link advertisements.

As described in Section 7.1, link state advertisements are uniquely identified by their type, link state ID, and advertising switch.

Link state advertisements are distributed throughout the switch fabric using a reliable flooding algorithm that ensures that all switches in the fabric are notified of any link state changes.

8.1 Originating Link State Advertisements

A new instance of each link state advertisement is originated any time the state of the switch or link changes. When a new instance of a link state advertisement is originated, its sequence number is incremented, its age is set to zero, and its checksum is calculated. The advertisement is then installed into the local link state database and forwarded out all fully operational interfaces (that is, those interfaces with a state greater than Waiting) for distribution throughout the switch fabric. See Section 8.2.4 for a description of the installation of the advertisement into the link state database and Section 8.2.5 for a description of how advertisements are forwarded.

A switch originates a new instance of a link state advertisement as a result of the following events:

- The state of one of the switch's interfaces changes such that the contents of the associated switch link advertisement changes.

- The designated switch on any of the switch's attached network links changes. The switch originates a new switch link advertisement. Also, if the switch itself is now the designated switch, it originates a new network link advertisement for the link.

- One of the switch's neighbor states changes to or from Full. If this changes the contents of the associated switch link advertisement, a new instance is generated. Also, if the switch is the designated switch for the attached network link, it originates a new network link advertisement for the link.

Two instances of the same link state advertisement must not be originated within the time period MinLSInterval. Note that this may require that the generation of the second instance to be delayed up to MinLSInterval seconds.

8.1.1 Switch Link Advertisements

A switch link advertisement describes the collected states of all functioning links attached to the originating switch -- that is, all attached links with an interface state greater than Down. A switch originates an empty switch link advertisement when it first becomes functional. It then generates a new instance of the advertisement each time one of its interfaces reaches a fully functioning state (Point-to-Point or better).

Each link in the advertisement is assigned a type, based on the state of interface, as shown in Table 4.

Interface state	Link type	Description
Point-to-Point	1	Point-to-point link
DS Other*	2	Multi-access link
Backup*	2	Multi-access link
DS**	2	Multi-access link

*If a full adjacency has been formed with the designated switch.

**If a full adjacency has been formed with at least one other switch on the link.

Table 4: Link Types in a Switch Link Advertisement

Each link in the advertisement is also assigned a link identifier based on its link type. In general, this value identifies another switch that also originates advertisements for the link, thereby providing a key for accessing other link state advertisements for the link. The relationship between link type and ID is shown in Table 5.

Type	Description	Link ID
1	Point-to-point link	Switch ID of neighbor switch
2	Multi-access link	Switch ID of designated switch

Table 5: Link IDs in a Switch Link Advertisement

In addition to a type and an identifier, the description of each link specifies the interface ID of the associated network link.

Finally, each link description includes the cost of sending a packet over the link. This output cost is expressed in the link state metric and must be greater than zero.

To illustrate the format of a switch link advertisement, consider the switch fabric shown in Figure 4.

In this example, switch SW1 has 5 neighboring switches (shown as boxes) distributed over 3 network links (shown as lines). The base MAC address of each switch is also shown adjacent to each box. On switch SW1, ports 01 and 02 attach to point-to-point network links,

while port 03 attaches to a multi-access network link with three attached switches. The interface state of each port is shown next to the line representing the corresponding link.

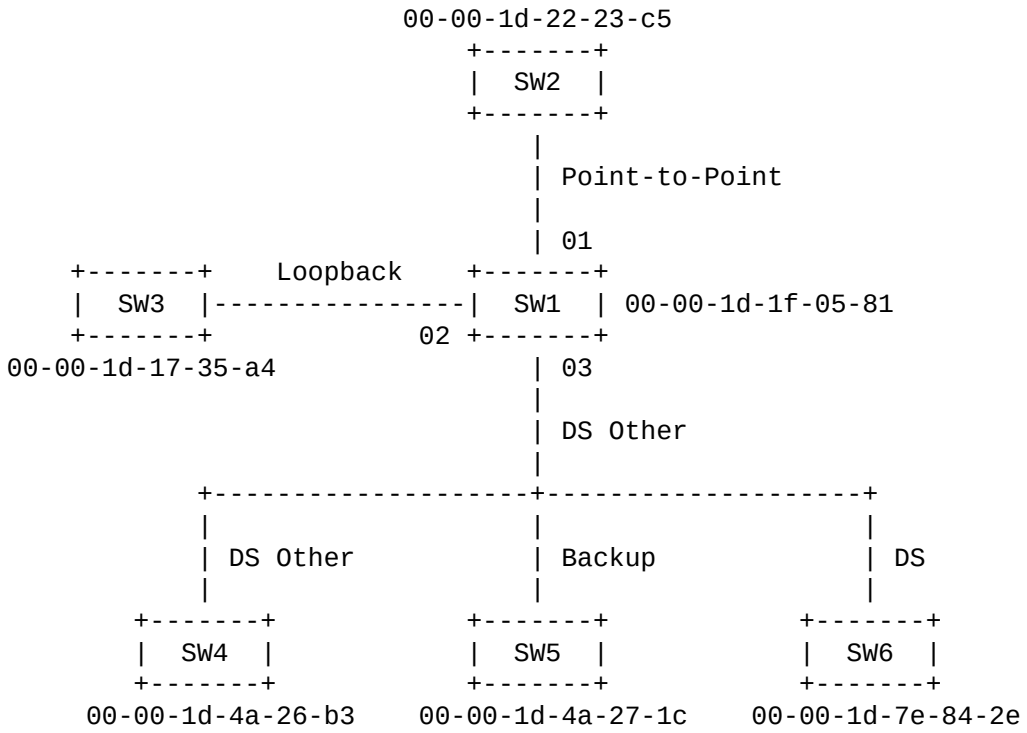


Figure 4: Sample Switch Fabric

The switch link advertisement generated by switch SW1 would contain the following data items:

```
; switch link advertisement for switch SW1

LS age = 0              ; always true on origination
Options = (T-bit|E-bit) ; options
LS type = 1             ; this is a switch link advert
```

```
                              ; SW1's switch ID
      Link State ID = 00-00-1d-1f-05-81-00-00-00-00
      Advertising switch = 00-00-1d-1f-05-81-00-00-00-00
      # links = 2

         ; link on interface port 1
         Link ID = 00-00-1d-22-23-c5-00-00-00-00    ; switch ID

         Link Data = 00-00-1d-1f-05-81-00-00-00-01  ; interface ID
         Type = 1                                   ; pt-to-pt link
         # other metrics = 0                        ; TOS 0 only
         TOS 0 metric = 1

         ; link on interface port 2 is not fully functional

         ; link on interface port 3
         Link ID = 00-00-1d-7e-84-2e-00-00-00-00    ; switch ID of DS
         Link Data = 00-00-1d-1f-05-81-00-00-00-03  ; interface ID
         Type = 2                                   ; multi-access
         # other metrics = 0                        ; TOS 0 only
         TOS 0 metric = 2
```

(See Section 11.2 for a detailed description of the format of a switch link advertisement.)

8.1.2 Network Link Advertisements

Network link advertisements are used to describe the switches attached to each multi-access network link.

Note: Network link advertisements are not generated for point-to-point links.

A network link advertisement is originated by the designated switch for the associated multi-access link once the switch has established a full adjacency with at least one other switch on the link. Each advertisement lists the switch IDs of those switches that are fully adjacent to the designated switch. The designated switch includes itself in this list.

To illustrate the format of a network link advertisement, consider again the switch fabric shown in Figure 4. In this example, network link advertisements will be generated only by switch SW6, the designated switch of the multi-access network link between switches SW1 and switches SW4, SW5, and SW6.

The network link advertisement generated by switch SW6 would contain the following data items:

```
; network link advertisement for switch SW6

LS age = 0                ; always true on origination
Options = (T-bit|E-bit)   ; options
LS type = 2               ; this is a network link advert

                            ; SW6's switch ID
Link State ID = 00-00-1d-73-84-2e-00-00-00-00
Advertising switch = 00-00-1d-73-84-2e-00-00-00-00

   Attached switch = 00-00-1d-7e-84-2e-00-00-00-00
   Attached switch = 00-00-1d-4a-26-b3-00-00-00-00
   Attached switch = 00-00-1d-1f-05-81-00-00-00-00
   Attached switch = 00-00-1d-4a-27-1c-00-00-00-00
```

(See Section 11.3 for a detailed description of the format of a network link advertisement.)

8.2 Distributing Link State Advertisements

Link state advertisements are distributed throughout the switch fabric encapsulated within Link State Update packets. A single Link State Update packet may contain several distinct advertisements.

To make the distribution process reliable, each advertisement must be explicitly acknowledged in a Link State Acknowledgment packet. Note, however, that multiple acknowledgments can be grouped together into a single Link State Acknowledgment packet. A sending switch retransmits unacknowledged Link State Update packets at regular intervals until they are acknowledged.

The remainder of this section is structured as follows:

- Section 8.2.1 presents an overview of the distribution process.
- Section 8.2.2 describes how an incoming Link State Update packet is processed.
- Section 8.2.3 describes how a Link State Packet is forwarded -- both by the originating switch and an intermediate receiving switch.
- Section 8.2.4 describes how advertisements are installed into the local database.
- Section 8.2.5 describes the retransmission of unacknowledged advertisements.

- Section 8.2.6 describes how advertisements are acknowledged.

8.2.1 Overview

The philosophy behind the distribution of link state advertisements is based on the concept of adjacencies -- that is, each switch is only required to remain synchronized with its adjacent neighbors.

When a switch originates a new instance of a link state advertisement, it formats the advertisement into a Link State Update packet and floods the packet out each fully operational interface -- that is, each interface with a state greater than Waiting. However, only those neighbors that are adjacent to the sending switch need to process the packet.

The sending switch indicates which of its neighbor switches should process the advertisement by specifying a particular multicast destination in the network layer address information (see Section 10.3). The sending switch sets the value of the network layer destination switch ID field according to the state of the interface over which the packet is sent:

- If the interface state is Point-to-Point, DS, or Backup, the switch is adjacent to all other switches on the link and all neighboring switches must process the packet. Therefore, the destination field is set to the multicast switch ID AllSPFSwitches.

- If the interface state is DS Other, the switch is only adjacent to the designated switch and the backup designated switch and only those two neighboring switches must process the packet. Therefore, the destination field is set to the multicast switch ID AllDSwitches.

A similar logic is used when a switch receives a Link State Update packet containing a new instance of a link state advertisement. After processing and acknowledging the packet, the receiving switch forwards the Link State Update packet as

- On the interface over which the original Link State Update packet was received:

- If the receiving switch is the designated switch for the attached network link, the packet is forwarded to all other switches on the link. (The destination field is set to AllSPFSwitches.) The originating switch will recognize that it was the advertisement originator and discard the packet.

- If the receiving switch is not the designated switch for the attached network link, the packet is not sent back out the interface over which it was received.

- On all other interfaces:

  - If the receiving switch is the designated switch for the attached network link, the packet is forwarded to all switches on the link. (The destination field is set to AllSPFSwitches.)

  - If the receiving switch is neither the designated switch or the backup designated switch for the attached network link, the packet is forwarded only to the designated switch and the backup designated switch. (The destination field is set to AllDSwitches.)

Each Link State Update packet is forwarded and processed in this fashion until all switches in the fabric have received notification of the new instance of the link state advertisement.

8.2.2 Processing an Incoming Link State Update Packet

When the a Link State Update packet is received, it is first subjected to a number of consistency checks. In particular, the Link State Update packet is associated with a specific neighbor. If the state of that neighbor is less than Exchange, the entire Link State Update packet is discarded.

Each link state advertisement contained in the packet is processed as follows:

1. Validate the advertisement's link state checksum and type. If the checksum is invalid or the type is unknown, discard the advertisement without acknowledging it.

2. If the advertisement's age is equal to MaxAge and there is currently no instance of the advertisement in the local link state database, then do the following:

a) Acknowledge the advertisement by sending a Link State Acknowledgment packet to the sending neighbor (see Section 8.2.6).

b) Purge all outstanding requests for equal or previous instances of the advertisement from the sending neighbor's Link State Request list.

c) If the neighbor is Exchange or Loading, install the advertisement in the link state database (see Section 8.2.4). Otherwise, discard the advertisement.

3. If the advertisement's age is equal to MaxAge and there is an instance of the advertisement in the local link state database, then do the following:

a) If the advertisement is listed in the link state retransmission list of any neighbor, remove the advertisement from the retransmission list(s) and delete the database copy of the advertisement.

b) Discard the received (MaxAge) advertisement without acknowledging it.

4. If the advertisement's age is less than MaxAge, attempt to locate an instance of the advertisement in the local link state database. If there is no database copy of this advertisement, or the received advertisement is more recent than the database copy (see Section 7.1.1), do the following:

a) If there is already a database copy, and if the database copy was installed less than MinLSInterval seconds ago, discard the new advertisement without acknowledging it.

b) Otherwise, forward the new advertisement out some subset of the local interfaces (see Section 8.2.3). Note whether the advertisement was sent back out the receiving interface for later use by the acknowledgment process.

c) Remove the current database copy from the Link state retransmission lists of all neighbors.

d) Install the new advertisement in the link state database, replacing the current database copy. (Note that this may cause the calculation of the set of best paths to be scheduled. See Section 9.) Timestamp the new advertisement with the time that it was received to prevent installation of another instance within MinLSInterval seconds.

e) Acknowledge the advertisement, if necessary, by sending a Link State Acknowledgment packet back out the receiving interface. (See Section 8.2.6.)

f) If the link state advertisement was initially advertised by the local switch itself, advance the advertisement sequence number and issue a new instance of the advertisement. (Receipt of a newer instance of an advertisement means that the local copy of the advertisement is left over from before the last time the switch was restarted.)

5. If the received advertisement is the same instance as the database copy (as determined by the algorithm described in Section 7.1.1), do the following:

a) If the advertisement is listed in the neighbor's link state retransmission list, the local switch is expecting an acknowledgment for this advertisement. Treat the received advertisement as an implied acknowledgment, and remove the advertisement from the link state retransmission list. Note this implied acknowledgment for later use by the acknowledgment process (Section 8.2.6).

b) Acknowledge the advertisement, if necessary, by sending a Link State Acknowledgment packet back out the receiving interface. (See Section 8.2.6.)

If the database copy of the advertisement is more recent than the instance just received, do the following:

a) Determine whether the instance is listed in the neighbor link state request list. If so, an error has occurred in the database exchange process. Restart the database exchange process by generating a neighbor BadLSReq event for the sending neighbor and terminate processing of the Link State Update packet.

b) Otherwise, generate an unusual event to network management and discard the advertisement.

8.2.3 Forwarding Link State Advertisements

When a new instance of an advertisement is originated or after an incoming advertisement has been processed, the switch must decide over which interfaces and to which neighbors the advertisement will be forwarded. In some instances, the switch may decide not to forward the advertisement over a particular interface because it is able to determine that the neighbors on that attached link have or

will receive the advertisement from another switch on the link.

The decision of whether to forward an advertisement over each of the switch's interfaces is made as follows:

1. Each neighboring switch attached to the interface is examined to determine whether it should receive and process the new advertisement. For each neighbor, the following steps are executed:

   a) If the neighbor state is less than Exchange, the neighbor need not receive or process the new advertisement.

   b) If the neighbor state is Exchange or Loading, examine the link state request list associated with the neighbor. If an instance of the new advertisement is on the list, the neighboring switch already has an instance of the advertisement. Compare the new advertisement to the neighbor's copy:

      o If the new advertisement is less recent, the neighbor need not receive or process the new advertisement.

      o If the two copies are the same instance, delete the advertisement from the link state request list. The neighbor need not receive or process the new advertisement [7].

      o Otherwise, the new advertisement is more recent. Delete the advertisement from the link state request list. The neighbor may need to receive and process the new advertisement.

   c) If the new advertisement was received from this neighbor, the neighbor need not receive or process the advertisement.

   d) Add the new advertisement to the link state retransmission list for the neighbor.

2. The switch must now decide whether to forward the new advertisement out the interface.

   a) If the link state advertisement was not added to any of the link state retransmission lists for neighbors attached to the interface, there is no need to forward the advertisement out the interface.

b) If the new advertisement was received on this interface, and it was received from either the designated switch or the backup designated switch, there is no need to forward the advertisement out the interface. Chances are all neighbors on the attached network link have also received the advertisement already.

c) If the new advertisement was received on this interface and the state of the interface is Point-to-Point, there is no need to forward the advertisement since the received advertisement was originated by the neighbor switch.

d) If the new advertisement was received on this interface, and the interface state is Backup -- that is, the switch itself is the backup designated switch -- there is no need to forward the advertisement out the interface. The designated switch will distribute advertisements on the attached network link.

e) Otherwise, the advertisement must be forwarded out the interface.

To forward a link state advertisement, the switch first increments the advertisement's age by InfTransDelay seconds to account for the transmission time over the link. The switch then copies the advertisement into a Link State Update packet

Forwarded advertisements are sent to all adjacent switches associated with the interface. If the interface state is Point-to-Point, DS, or Backup, the destination switch ID field of the network layer address information is set to the multicast switch ID AllSPFSwitches. If the interface state is DS Other, the destination switch ID field is set to the multicast switch ID AllDSwitches.

8.2.4 Installing Link State Advertisements in the Database

When a new link state advertisement is installed into the link state database, as the result of either originating or receiving a new instance of an advertisement, the switch must determine whether the best paths need to be recalculated. To make this determination, do the following:

1. Compare the contents of the new instance with the contents of the old instance (assuming the older instance is available). Note that this comparison does not include any data from the link state header. Differences in fields within the header (such as the sequence number and checksum, which are guaranteed to be different in different instances of an advertisement) are of no consequence

when deciding whether or not to recalculate the set of best paths.

2. If there are no differences in the contents of the two advertisement instances, there is no need to recalculate the set of best paths.

3. Otherwise, the set of best paths must be recalculated.

Note also that the older instance of the advertisement must be removed from the link state database when the new advertisement is installed. The older instance must also be removed from the link state retransmission lists of all neighbors.

8.2.5 Retransmitting Link State Advertisements

When a switch sends a link state advertisement to an adjacent neighbor, it records the advertisement in the neighbor's link state retransmission list. To ensure the reliability of the distribution process, the switch continues to periodically retransmit the advertisements specified in the list until they are acknowledged.

The interval timer used to trigger retransmission of the advertisements is set to RxmtInterval seconds, as found in the interface data structure. Note that if this value is too low, needless retransmissions will ensue. If the value is too high, the speed with which the databases synchronize across adjacencies may be affected if there are lost packets.

When the interval timer expires, entries in the retransmission list are formatted into one or more Link State Update packets. (Remember that multiple advertisements can fit into a single Link State Update packet.) The age field of each advertisement is incremented by InfTransDelay, as found in the interface data structure, before the advertisement is copied into the outgoing packet.

Link State Update packets containing retransmitted advertisements are always sent directly to the adjacent switch. That is, the destination field of the network layer addressing information is set to the switch ID of the neighboring switch.

If the adjacent switch goes down, retransmissions will continue until the switch failure is detected and the adjacency is torn down by the VLSP discovery process. When the adjacency is torn down, the link state retransmission list is cleared.

8.2.6 Acknowledging Link State Advertisements

Each link state advertisement received by a switch must be acknowledged. In most cases, this is done by sending a Link State Acknowledgment packet. However, acknowledgments can also be done implicitly by sending Link State Update packets (see step 4a of Section 8.2.2).

Multiple acknowledgments can be grouped together into a single Link State Acknowledgment packet.

Sending an acknowledgment

Link State Acknowledgment packets are sent back out the interface over which the advertisement was received. The packet can be sent immediately to the sending neighbor, or it can be delayed and sent when an interval timer expires.

- Sending delayed acknowledgments facilitates the formatting of multiple acknowledgments into a single packet. This enables a single packet to send acknowledgments to several neighbors at once by using a multicast switch ID in the destination field of the network layer addressing information (see below). Delaying acknowledgments also randomizes the acknowledgment packets sent by the multiple switches attached to a multi-access network link.

  Note that the interval used to time delayed acknowledgments must be short (less than RxmtInterval) or needless retransmissions will ensue.

  Delayed acknowledgments are sent to all adjacent switches associated with the interface. If the interface state is Point-to-Point, DS, or Backup, the destination field of the network layer addressing information is set to the multicast switch ID AllSPFSwitches. If the interface state is DS Other, the destination field is set to the multicast switch ID AllDSwitches.

- Immediate acknowledgments are sent directly to a specific neighbor in response to the receipt of duplicate link state advertisements. These acknowledgments are sent immediately when the duplicate is received.

The method used to send a Link State Acknowledgment packet -- either delayed or immediate -- depends on the circumstances surrounding the receipt of the advertisement, as shown in Table 6. Note that switches with an interface state of Backup send

acknowledgments differently than other switches because they play a slightly different role in the distribution process (see Section 8.2.3).

Circumstances	Action taken in state: Backup	Other states
Advertisement was forwarded back out receiving interface	No ack sent	No ack sent
Advertisement is more recent than database copy, but was not forwarded back out receiving interface	Delayed ack sent if advertisement received from DS, else do nothing	Delayed ack sent
Advertisement was a duplicate treated as an implied acknowledgment (step 4a of Section 8.2.2)	Delayed ack sent if advertisement received from DS, else do nothing	No ack sent
Advertisement was a duplicate not treated as an implied acknowledgment	Immediate ack sent	Immediate ack sent
Advertisement age equal to MaxAge and no current instance found in database	Immediate ack sent	Immediate ack sent

Table 6: Sending Link State Acknowledgments

Receiving an acknowledgment

When the a Link State Acknowledgment packet is received, it is first subjected to a number of consistency checks. In particular, the packet is associated with a specific neighbor. If the state of that neighbor is less than Exchange, the entire Link State Acknowledgment packet is discarded.

Each acknowledgment contained in the packet is processed as follows:

- If the advertisement being acknowledged has an instance in the link state retransmission list for the sending neighbor, do the following:
  - If the acknowledgment is for the same instance as that specified in the list (as determined by the procedure described in Section 7.1.1), remove the instance from the retransmission list.
  - Otherwise, log the acknowledgment as questionable.

8.3 Aging the Link State Database

Each link state advertisement has an age field, containing the advertisement's age, expressed in seconds. When the advertisement is copied into a Link State Update packet for forwarding out a particular interface, the age is incremented by InfTransDelay seconds to account for the transmission time over the link. An advertisement's age is never incremented past the value MaxAge. Advertisements with an age of MaxAge are not used to calculate best paths.

If a link state advertisement's age reaches MaxAge, the switch flushes the advertisement from the switch fabric by doing the following:

- Originate a new instance of the advertisement with the age field set to MaxAge. The distribution process will eventually result in the advertisement being removed from the retransmission lists of all switches in the fabric.
- Once the advertisement is no longer contained in the link state retransmission list of any neighbor and no neighbor is in a state of Exchange or Loading, remove the advertisement from the local link state database.

8.3.1 Premature Aging of Advertisements

A link state advertisement can be prematurely flushed from the switch fabric by forcing its age to MaxAge and redistributing the advertisement.

A switch that was previously the designated switch for a multi-access network link but has lost that status due to a failover to the backup designated switch prematurely ages the network link advertisements it originated for the link.

Premature aging also occurs when an advertisement's sequence number must wrap -- that is, when the current advertisement instance has a sequence number of 0x7fffffff. In this circumstance, the advertisement is prematurely aged so that the next instance of the advertisement can be originated with a sequence number of 0x80000001 and be recognized as the most recent instance.

A switch may only prematurely age those link state advertisements for which it is the advertising switch.

9. Calculating the Best Paths

Once an adjacency has been formed and the two switches have synchronized their databases, each switch in the adjacency calculates the best path(s) to all other switches in the fabric, using itself as the root of each path. In this context, "best" path means that path with the lowest total cost metric across all hops. If there are multiple paths with the same (lowest) total cost metric, they are all calculated. Best paths are stored in the area data structure.

Paths are calculated using the well-known Dijkstra algorithm. For a detailed description of this algorithm, the reader is referred to [Perlman], or any of a number of standard textbooks dealing with network routing.

Note that whenever there is a change in an adjacency relationship, or any change that alters the topology of the switch fabric, the set of best paths must be recalculated.

10. Protocol Packets

This section describes VLS protocol packets and link state advertisements.

There are five distinct VLSP packet types, as listed in Table 7.

Type	Packet Name	Function	Description
1	Hello	Select DS/Backup DS	Section 10.6.1
2	Database Description	Summarize database contents	Section 10.6.2
3	LS Request	Database download	Section 10.6.3
4	LS Update	Database update	Section 10.6.4
5	LS Ack	Flooding acknowledgment	Section 10.6.5

Table 7: VLSP Packet Types

All VLSP packets are encapsulated within a standard ISMP packet, with the VLS packet carried in the ISMP message body. The ISMP packet is described in Section 10.1.

Since it is important that the link state databases remain synchronized throughout the switch fabric, processing of both incoming and outgoing routing protocol packets should take priority over ordinary data packets. Section 10.2 describes packet processing.

All VLSP packets begin with network layer addressing information, described in Section 10.3, followed by a standard header, described in Section 10.4.

With the exception of Hello packets, all VLSP packets deal with lists of link state advertisements. The format of a link state advertisement is described in Section 11.

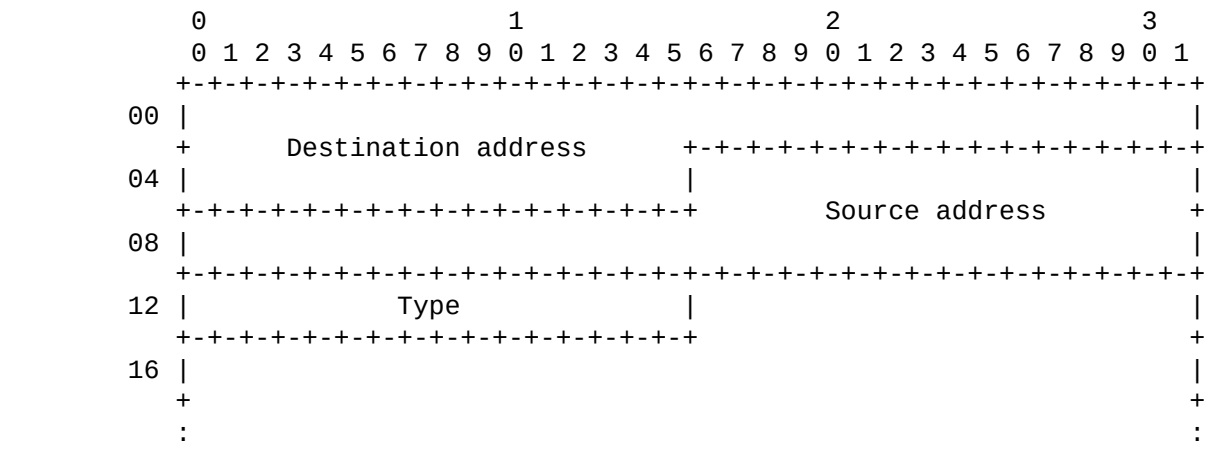
10.1 ISMP Packet Format

All VLSP packets are encapsulated within a standard ISMP packet. ISMP packets are of variable length and have the following general structure:

- Frame header
- ISMP packet header
- ISMP message body

10.1.1 Frame Header

ISMP packets are encapsulated within an IEEE 802-compliant frame using a standard header as shown below:



Destination address

This 6-octet field contains the Media Access Control (MAC) address of the multicast channel over which all switches in the fabric receive ISMP packets. The destination address of all ISMP packets contain a value of 01-00-1D-00-00-00.

Source address

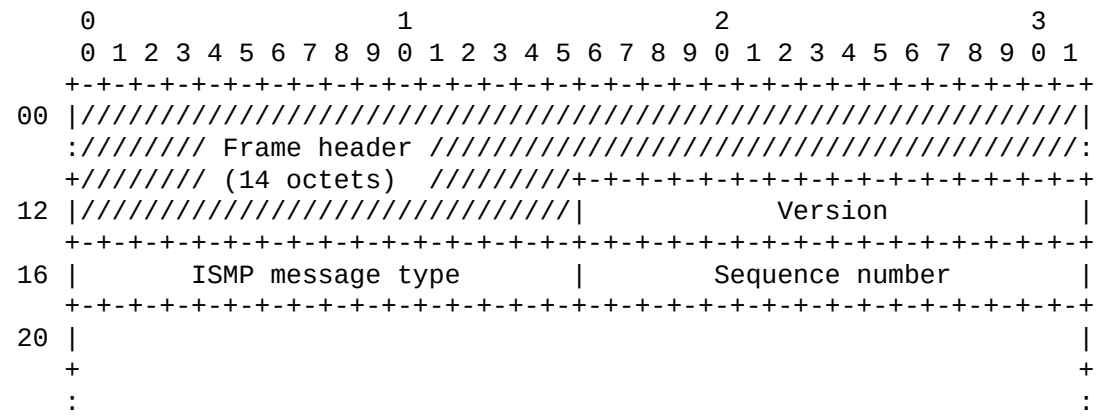
This 6-octet field contains the physical (MAC) address of the switch originating the ISMP packet.

Type

This 2-octet field identifies the type of data carried within the frame. The type field of ISMP packets contains the value 0x81FD.

10.1.2 ISMP Packet Header

The ISMP packet header consists of 6 octets, as shown below:



Frame header

This 14-octet field contains the frame header.

Version

This 2-octet field contains the version number of the InterSwitch Message Protocol to which this ISMP packet adheres. This document describes ISMP Version 2.0. ISMP message type

This 2-octet field contains a value indicating which type of ISMP message is contained within the message body. Valid values are as follows:

1	(reserved)
2	Interswitch Keepalive messages
3	Interswitch Link State messages
4	Interswitch Spanning Tree BPDU messages and Interswitch Remote Blocking messages
5	Interswitch Resolve and New User messages
6	(reserved)
7	Tag-Based Flood messages
8	Interswitch Tap messages

All VLS protocol messages have an ISMP message type of 3.

Sequence number

This 2-octet field contains an internally generated sequence number used by the various protocol handlers for internal synchronization of messages.

10.1.3 ISMP Message Body

The ISMP message body is a variable-length field containing the actual data of the ISMP message. The length and content of this field are determined by the value found in the message type field. VLSP packets are contained in the ISMP message body.

10.2 VLSP Packet Processing

Note that with the exception of Hello packets, VLSP packets are sent only between adjacent neighbors. Therefore, all packets travel a single hop.

VLSP does not support fragmentation and reassembly of packets. Therefore, packets containing lists of link state advertisements or advertisement headers must be formatted such that they contain only as many advertisements or headers as will fit within the size constraints of a standard ethernet frame.

When a protocol packet is received by a switch, it must first pass the following criteria before being accepted for further processing:

- The checksum number must be correct.
- The destination switch ID (as found in the network layer address information) must be the switch ID of the receiving switch, or one of the multicast switch IDs AllSPFSwitches or AllDSwitches.

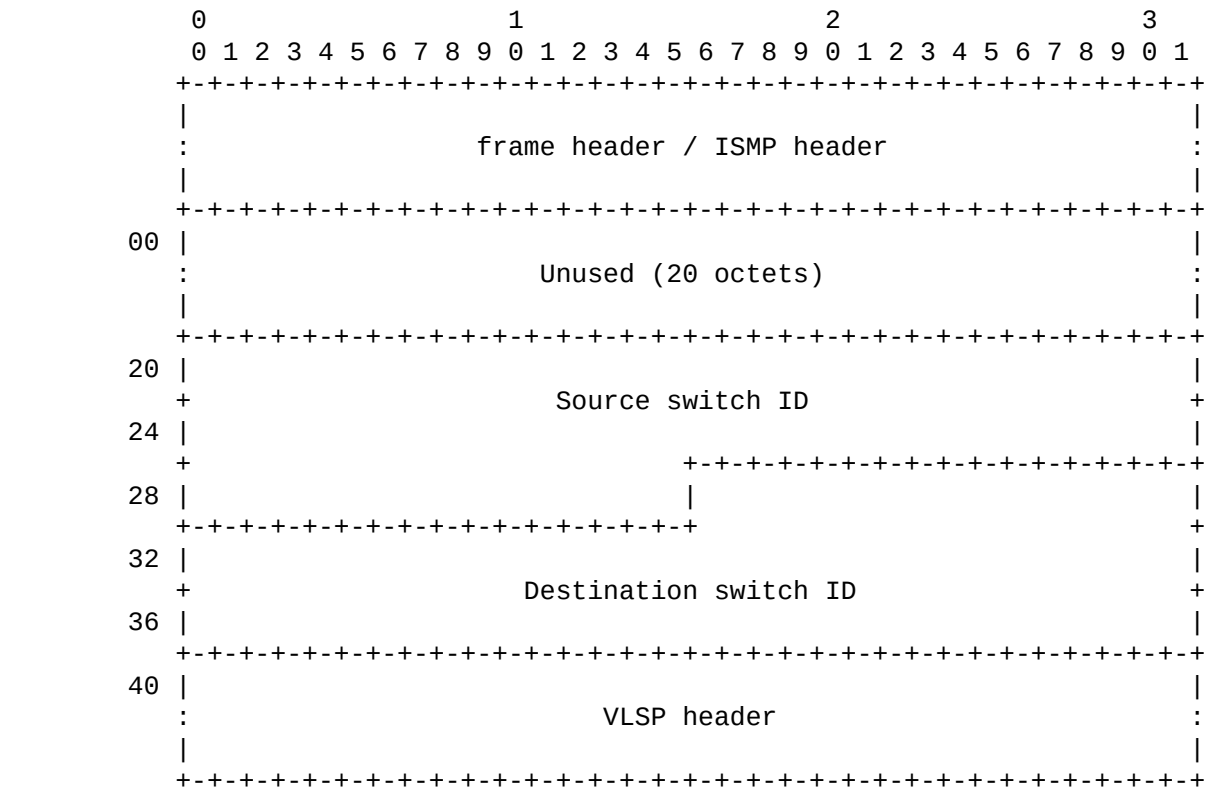
  If the destination switch ID is the multicast switch ID AllDSwitches, the state of the receiving interface must be Point-to-Point, DS, or Backup.
- The source switch ID (as found in the network layer address information) must not be that of the receiving switch. (That is, locally originated packets should be discarded.)

At this point, if the packet is a Hello packet, it is accepted for further processing.

Since all other packet types are only sent between adjacent neighbors, the packet must have been sent by one of the switch's active neighbors. If the source switch ID matches the switch ID of one of the receiving switch's active neighbors (as stored in the interface data structure associated with the inport interface), the packet is accepted for further processing. Otherwise, the packet is discarded.

10.3 Network Layer Address Information

As mentioned in Section 2.2.1, portions of the VLS protocol (as derived from OSPF) are dependent on certain network layer addresses -- in particular, the AllSPFSwitches and AllDSwitches multicast addresses that drive the distribution of link state advertisements throughout the switch fabric. In order to facilitate the implementation of the protocol at the physical MAC layer, network layer address information is encapsulated in the VSLP packets. This information immediately follows the ISMP frame and packet header and immediately precedes the VLSP packet header, as shown below:



Source switch ID

This 10-octet field contains the switch ID of the sending switch.

Destination switch ID

This 10-octet field contains the switch ID of the packet destination. The value here is set as follows:

- Hello packets are addressed to the multicast switch ID AllSPFSwitches.
- The designated switch and the backup designated switch address initial Link State Update packets and Link State Acknowledgment packets to the multicast switch ID AllSPFSwitches.
- All other switches address initial Link State Update packets and Link State Acknowledgment packets to the multicast switch ID AllDSwitches.
- Retransmissions of Link State Update packets are always addressed directly to the nonresponding switch.
- Database Description packets and Link State Request are always addressed directly to the other switch participating in the database exchange process.

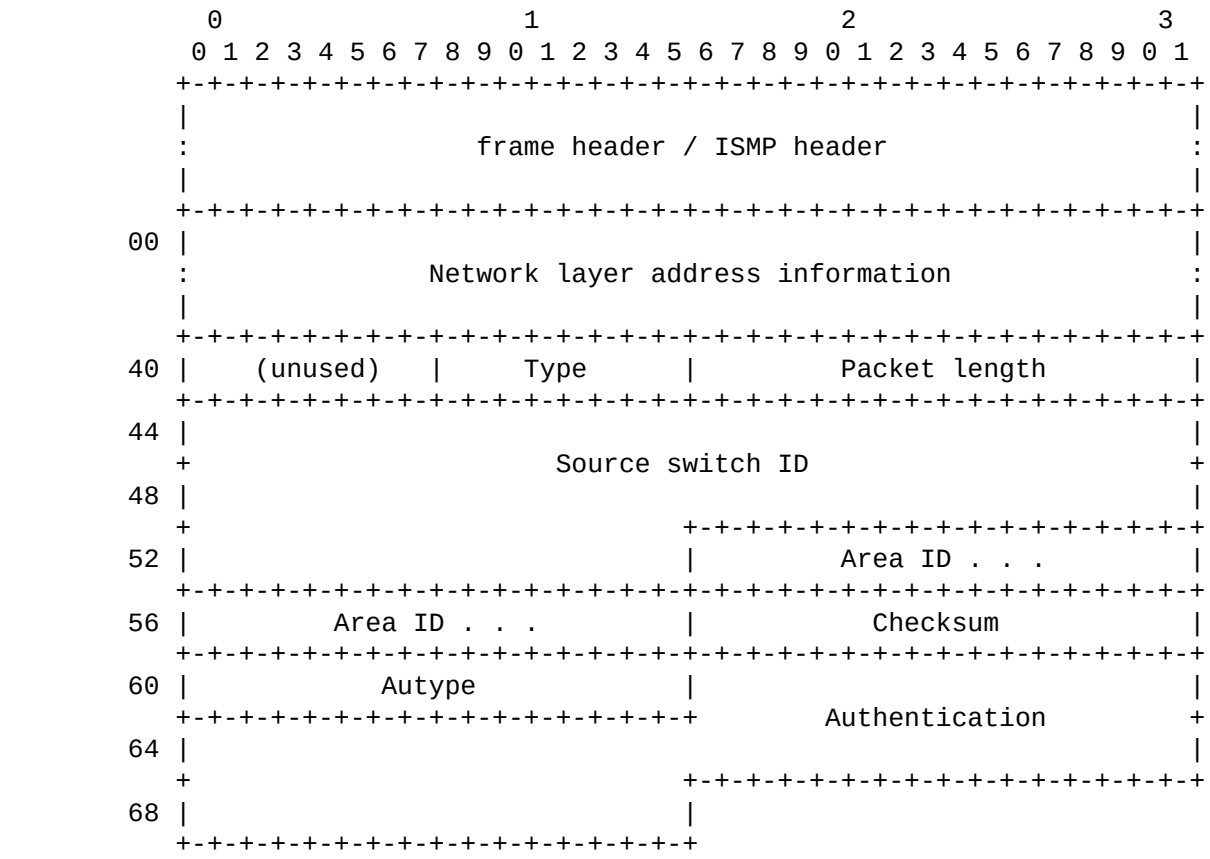
VLSP header

This 30-octet field contains the VLSP standard header. See Section 10.4.

10.4 VLSP Packet Header

Every VLSP packet starts with a common 30-octet header. This header, along with the data found in the network layer address information, contains all the data necessary to determine whether the packet should be accepted for further processing. (See Section 10.1.)

The format of the VLSP header is shown below. Note that the header starts at offset 36 of the ISMP message body, following the network layer address information.



Type

This 1-octet field contains the packet type. Possible values are as follows:

1 Hello
2 Database Description
3 Link State Request
4 Link State Update
5 Link State Acknowledgment

Packet length

This 2-octet field contains the length of the protocol packet, in bytes, calculated from the start of the VLSP header, at offset 20 of the ISMP message body. If the packet length is not an integral number of 16-bit words, the packet is padded with an octet of zero (see the description of the checksum field, below).

Switch ID

This 10-octet field contains the switch ID of the sending switch.

Area ID

This 4-octet field contains the area identifier. Since VLSP does not support multiple areas, the value here is always zero.

Checksum

This 2-octet field contains the packet checksum value. The checksum is calculated as the 16-bit one's complement of the one's complement sum of all the 16-bit words in the packet, beginning with the VLSP header, excluding the authentication field. If the packet length is not an integral number of 16-bit words, the packet is padded with an octet of zero before calculating the checksum.

AuType

This 2-octet field identifies the authentication scheme to be used for the packet. Since authentication is not supported by this version of VLSP, this field contains zero.

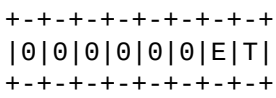
Authentication

This 8-octet field is reserved for use by the authentication scheme. Since authentication is not supported by this version of VLSP, this field contains zeroes.

10.5 Options Field

Hello packets and Database Description packets, as well as link state advertisements, contain a 1-octet options field. Using this field, a switch can communicate its optional capabilities to other VLSP switches. The receiving switch can then choose whether or not to support those optional capabilities. Thus, switches of differing capabilities potentially can be mixed within a single VLSP routing domain.

Two optional capabilities are currently defined in the options field: routing based on Type of Service (TOS) and support for external routing beyond the local switch fabric. These two capabilities are specified in the options field as shown below.



The options field

T-bit

The T-bit specifies the switch's Type of Service (TOS) capability. If the T-bit is set, the switch supports routing based on nonzero types of service.

E-bit

The E-bit specifies the switch's external routing capability. If the E-bit is set, the switch supports external routing.

Note: The current version of VLSP supports neither of these capabilities. Therefore, both the T-bit and the E-bit are clear and the options field contains a value of zero.

10.6 Packet Formats

This section contains detailed descriptions of the five VLS protocol packets.

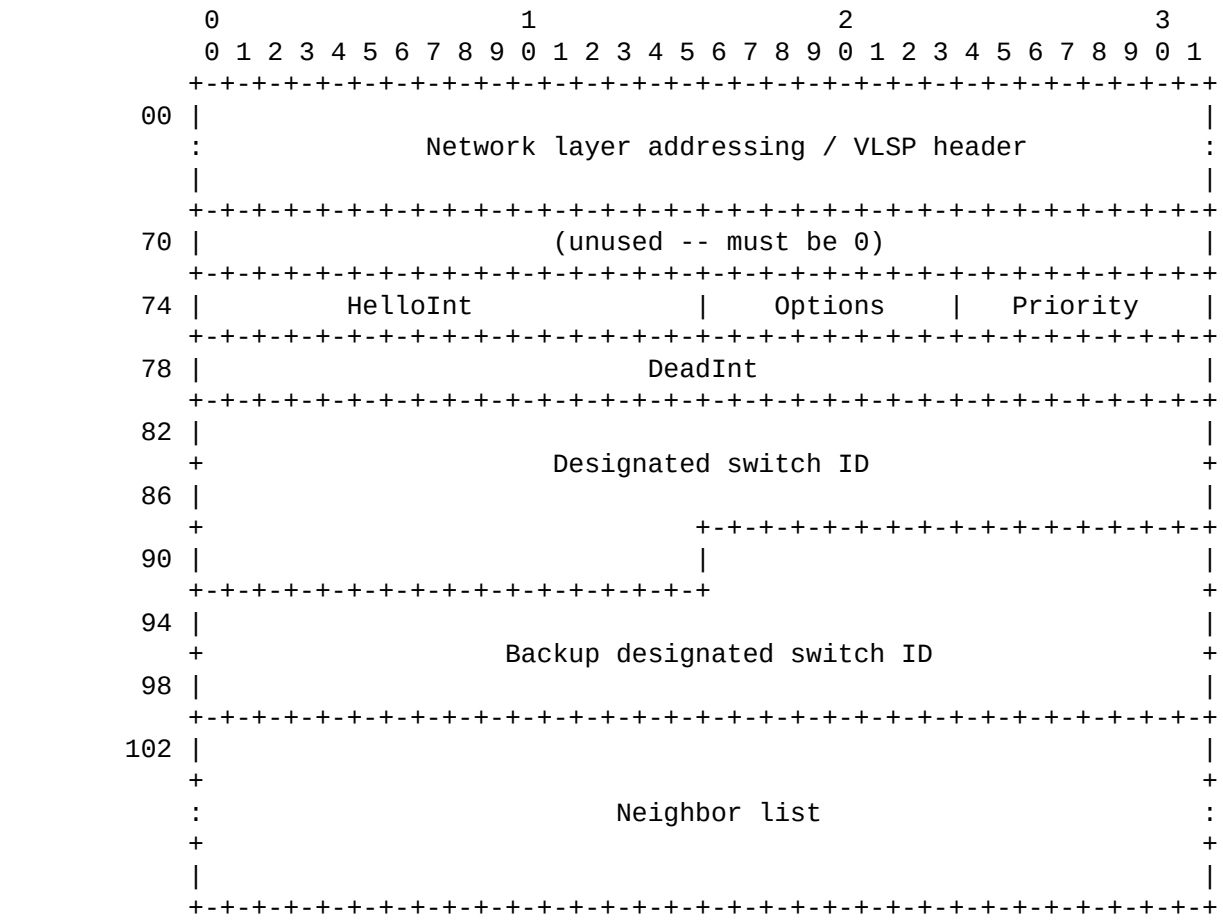
10.6.1 Hello Packets

Hello packets are sent periodically over multi-access switch interfaces in order to discover and maintain neighbor relationships.

Note: Hello packets are not sent over point-to-point network links. For point-to-point links, the VLS protocol relies on the VlanHello protocol [IDhello] to notify it of neighboring switches.

Since all switches connected to a common network link must agree on certain interface parameters, these parameters are included in each Hello packet. A switch receiving a Hello packet that contains parameters inconsistent with its own view of the interface will not establish a neighbor relationship with the sending switch.

The format of a Hello packet is shown below.



Network layer addressing / VLSP header

This 70-octet field contains the network layer addressing information and the standard VLS protocol packet header. The packet header type field contains a value of 1.

HelloInt

This 2-octet field contains the interval, in seconds, at which this switch sends Hello packets.

Options

This 1-octet field contains the optional capabilities supported by the switch, as described in Section 10.5.

Priority

This 1-octet field contains the switch priority used in selecting the designated switch and backup designated switch (see Section 6.3.1). If the value here is zero, the switch is ineligible to become the designated switch or the backup designated switch.

DeadInt

This 4-octet field contains the length of time, in seconds, that neighboring switches will wait before declaring the interface down once they stop receiving Hello packets over the interface. The value here is equal to the value of SwitchDeadInterval, as found in the interface data structure.

Designated switch

This 10-octet field contains the switch ID of the designated switch for this network link, as currently understood by the sending switch. This value is set to zero if the designated switch selection process has not yet begun.

Backup designated switch

This 10-octet field contains the switch ID of the backup designated switch for the network link, as currently understood by the sending switch. This value is set to zero if the backup designated switch selection process has not yet begun.

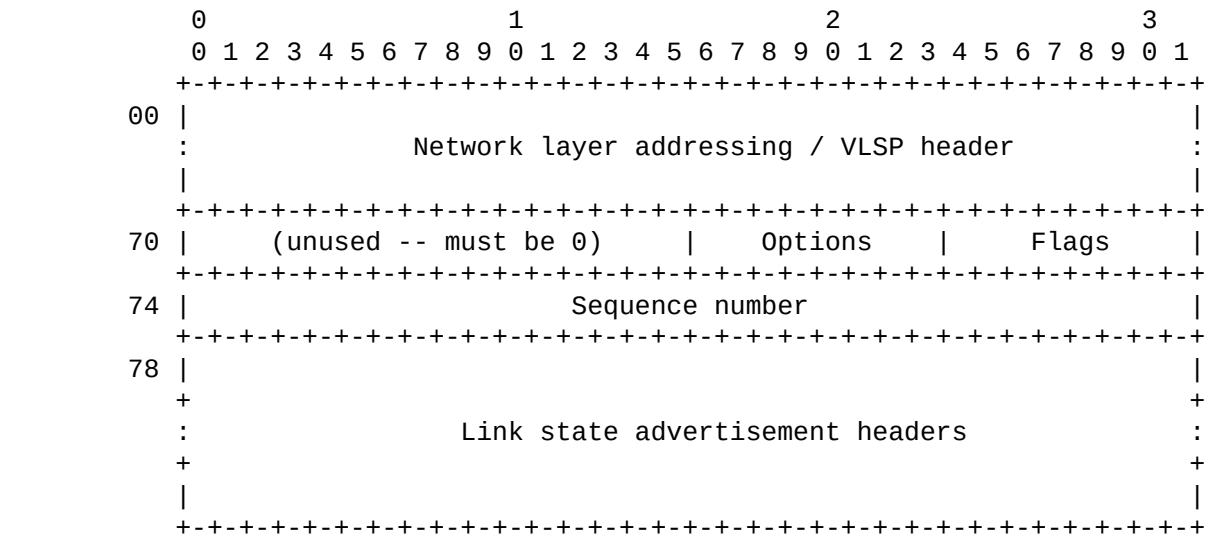
Neighbor list

This variable-length field contains a list of switch IDs of each switch from which the sending switch has received a valid Hello packet within the last SwitchDeadInterval seconds.

10.6.2 Database Description Packets

Database Description packets are exchanged while an adjacency is being formed between two neighboring switches and are used to describe the contents of the topological database. For a complete description of the database exchange process, see Section 7.2.

The format of a Database Description packet is shown below.



Network layer addressing / VLSP header

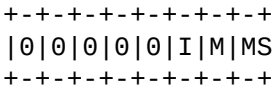
This 70-octet field contains the network layer addressing information and the standard VLS protocol packet header. The packet header type field contains a value of 2.

Options

This 1-octet field contains the optional capabilities supported by the switch, as described in Section 10.5.

Flags

This 1-octet field contains a set of bit flags that are used to coordinate the database exchange process. The format of this octet is as follows:



I-bit (Init)

The I-bit is used to signal the start of the exchange. It is set while the two switches negotiate the master/slave relationship and the starting sequence number.

M-bit (More)

The M-bit is set to indicate that more Database Description packets to follow.

MS-bit (Master/Slave)

The MS-bit is used to indicate which switch is the master of the exchange. If the bit is set, the sending switch is the master during the database exchange process. If the bit is clear, the switch is the slave.

Sequence number

This 4-octet field is used to sequence the Database Description packets during the database exchange process. The two switches involved in the exchange process agree on the initial value of the sequence number during the master/slave negotiation. The number is then incremented for each Database Description packet in the exchange.

To acknowledge each Database Description packet sent by the master, the slave sends a Database Description packet that echoes the sequence number of the packet being acknowledged.

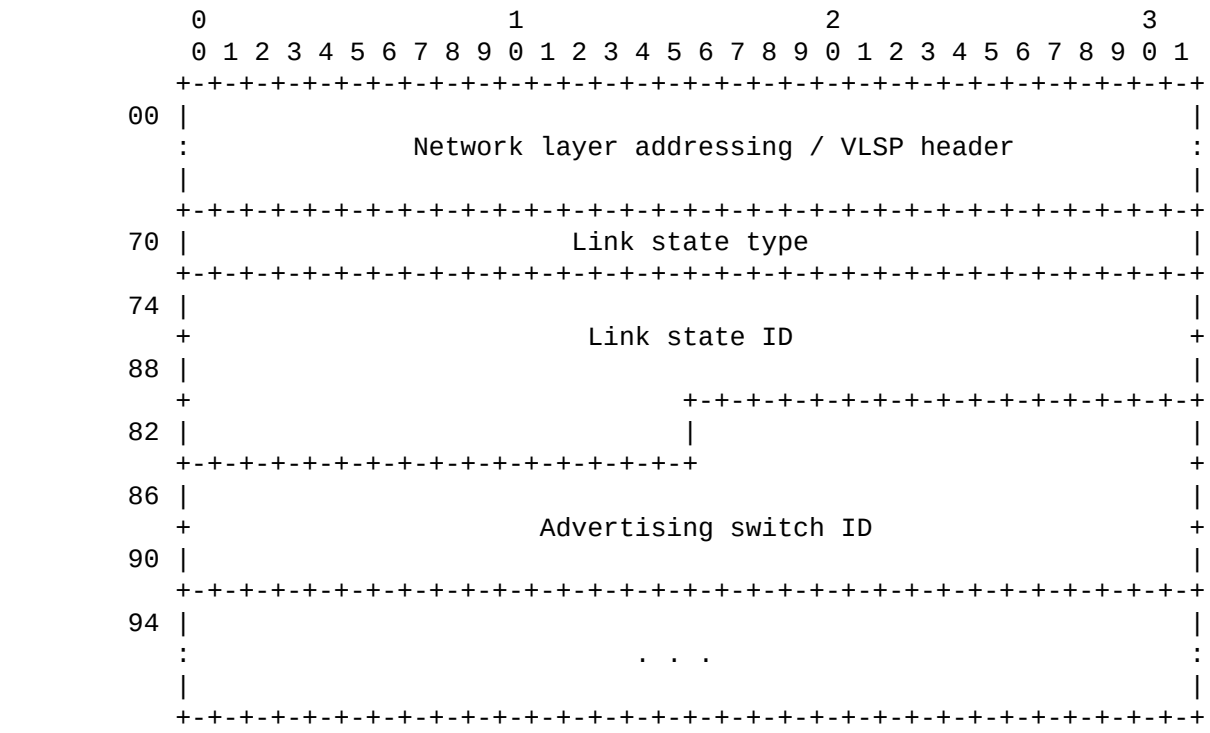
Link state advertisement headers

This variable-length field contains a list of link state headers that describe a portion of the master's topological database. Each header uniquely identifies a link state advertisement and its current instance. (See Section 11.1 for a detailed description of a link state advertisement header.) The number of headers included in the list is calculated implicitly from the length of the packet, as stored in the VLSP packet header (see Section 10.4).

10.6.3 Link State Request Packets

Link State Request packets are used to request those pieces of the neighbor's database that the sending switch has discovered (during the database exchange process) are more up-to-date than instances in its own database. Link State Request packets are sent as the last step in bringing up an adjacency. (See Section 7.3.)

The format of a Link State Request packet is shown below.



Network layer addressing / VLSP header

This 70-octet field contains the network layer addressing information and the standard VLS protocol packet header. The packet header type field contains a value of 3.

Link state type

This 4-octet field contains the link state type of the requested link state advertisement, as stored in the advertisement header.

Link state ID

This 10-octet field contains the link state ID of the requested link state advertisement, as stored in the advertisement header.

Advertising switch

This 10-octet field contains the switch ID of advertising switch for the requested link state advertisement, as stored in the advertisement header.

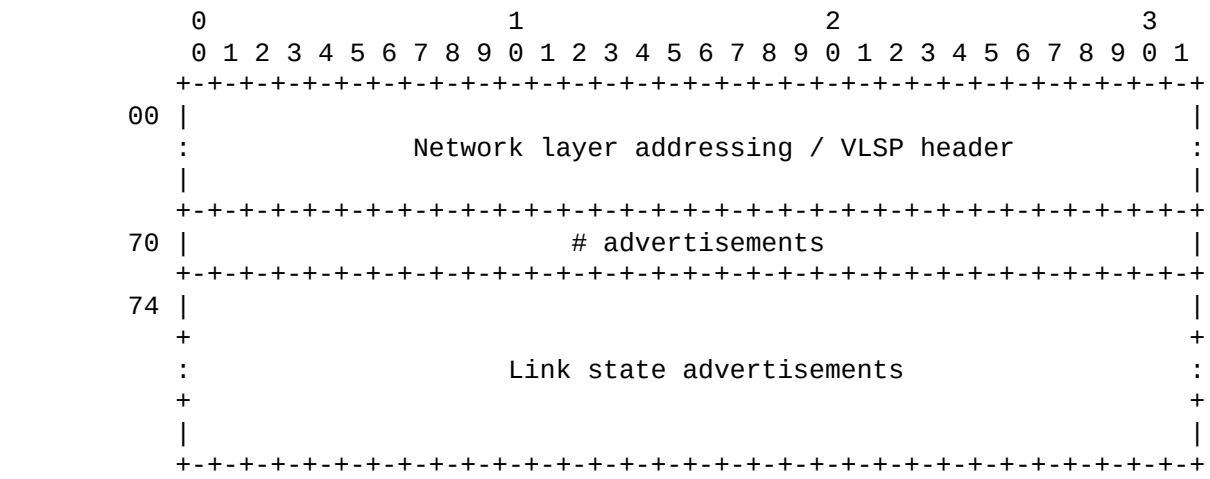
Note that the last three fields uniquely identify the advertisement, but not its instance. The receiving switch will respond with its most recent instance of the specified advertisement.

Multiple link state advertisements can be requested in a single Link State Request packet by repeating the link state type, ID, and advertising switch for each requested advertisement. The number of advertisements requested is calculated implicitly from the length of the packet, as stored in the VLSP packet header.

10.6.4 Link State Update Packets

Link State Update packets are used to respond to a Link State Request packet or to advertise a new instance of one or more link state advertisements. Link State Update packets are acknowledged with Link State Acknowledgment packets. For more information on the use of Link State Update packets, see Section 7 and Section 8.

The format of a Link State Update packet is shown below.



Network layer addressing / VLSP header

This 70-octet field contains the network layer addressing information and the standard VLS protocol packet header. The packet header type field contains a value of 4.

advertisements

This 4-octet field contains the number of link state advertisements included in the packet.

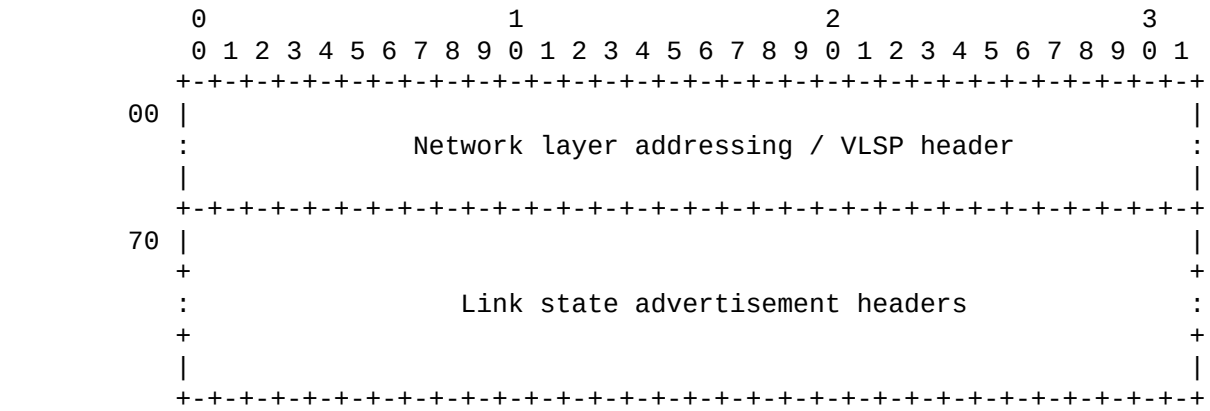
Link state advertisements

This variable-length field contains a list of link state advertisements. For a detailed description of the different types of link state advertisements, see Section 11.

10.6.5 Link State Acknowledgment Packets

Link State Acknowledgment Packets are used to explicitly acknowledge one or more Link State Update packets, thereby making the distribution of link state advertisements reliable. (See Section 8.2.6.)

The format of a Link State Acknowledgment packet is shown below.



Network layer addressing / VLSP header

This 70-octet field contains the network layer addressing information and the standard VLS protocol packet header. The packet header type field contains a value of 5.

Link state advertisement headers

This variable-length field contains a list of link state headers that are being acknowledged by this packet. Each header uniquely identifies a link state advertisement and its current instance. (See Section 11.1 for a detailed description of a link state advertisement header.) The number of headers included in the list is calculated implicitly from the length of the packet, as stored in the VLSP packet header (see Section 10.4).

11. Link State Advertisement Formats

Link state advertisements are used to describe various pieces of the routing topology within the switch fabric. Each switch in the fabric maintains a complete set of all link state advertisements generated throughout the fabric. (Section 8.1 describes the circumstances under which a link state advertisement is originated. Section 8.2 describes how advertisements are distributed throughout the switch fabric.) This collection of advertisements, known as the link state (or topological) database, is used to calculate a set of best paths to all other switches in the fabric.

There are two types of link state advertisement, as listed in Table 8.

Type	Name	Function	Description
1	Switch link advertisement	Lists all network linksattached to a switch	Section 11.2
2	Network link advertisement	Lists all adjacencies on a network link	Section 11.3

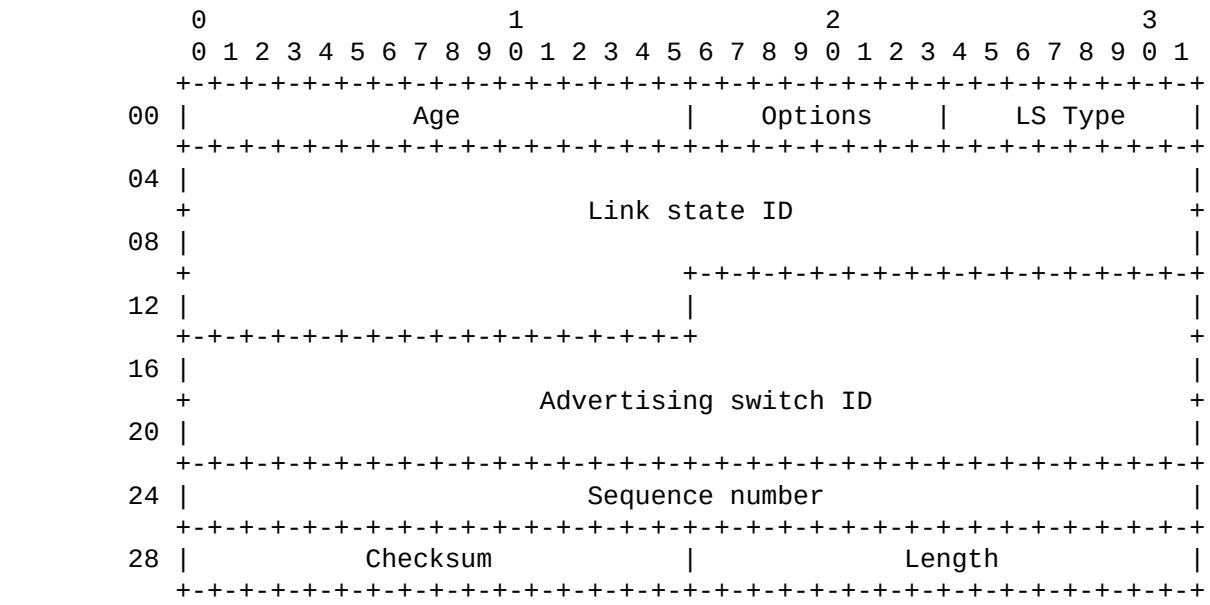
Table 8: Link State Advertisement Types

Each link state advertisement begins with a standard header, described in Section 11.1.

11.1 Link State Advertisement Headers

All link state advertisements begin with a common 32-octet header. This header contains information that uniquely identifies the advertisement -- its type, link state ID, and the switch ID of its advertising switch. Also, since multiple instances of a link state advertisement can exist concurrently in the switch fabric, the header contains information that permits a switch to determine which instance is the most recent -- the age, sequence number and checksum.

The format of the link state advertisement header is shown below.



Age

This 2-octet field contains the time, in seconds, since this instance of the link state advertisement was originated.

Options

This 1-octet field contains the optional capabilities supported by the advertising switch, as described in Section 10.5.

LS type

This 1-octet field contains the type of the link state advertisement. Possible values are:

1 Switch link advertisement
2 Network link advertisement

Link state ID

This 10-octet field identifies the switch that originates advertisements for the link. The content of this field depends on the advertisement's type.

o For a switch link advertisement, this field contains the switch ID of the originating switch

o For a network link advertisement, this field contains the switch ID of the designated switch for the link

Note: In VLSP, the link state ID of an advertisement is always the same as the advertising switch. This level of redundancy results from the fact that OSPF uses additional types of link state advertisements for which the originating switch is not the advertising switch.

Advertising switch

This 10-octet field contains the switch ID of the switch that originated the link state advertisement.

Sequence number

This 4-octet field is used to sequence the instances of a particular link state advertisement. The number is incremented for each new instance.

Checksum

This 2-octet field contains the checksum of the complete contents of the link state advertisement, excluding the age field. The checksum used is commonly referred to as the Fletcher checksum and is documented in [RFC905]. Note that since this checksum is calculated for each separate advertisement, a protocol packet containing lists of advertisements or advertisement headers will contain multiple checksum values.

Length

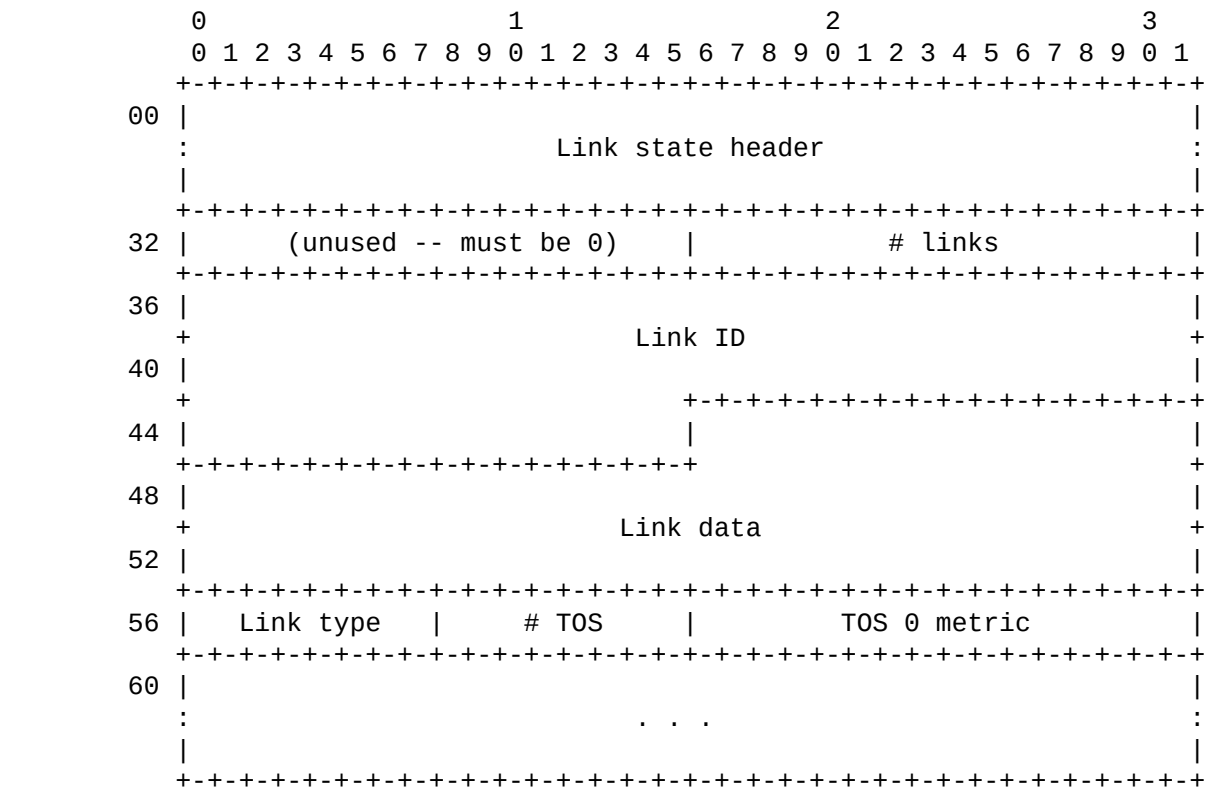
This 2-octet field contains the total length, in octets, of the link state advertisement, including the header.

11.2 Switch Link Advertisements

A switch link advertisement is used to describe all functioning network links of a switch, including the cost of using each link.

Each functioning switch in the fabric originates one, and only one, switch link advertisement -- all of the switch's links must be described in a single advertisement. A switch originates its first switch link advertisement (containing no links) when it first becomes functional. It then originates a new instance of the advertisement each time any of its neighbor states changes such that the contents of the advertisement changes. See Section 8.1 for details on originating a switch link advertisement.

The format of a switch link advertisement is shown below.



Link state header

This 32-octet field contains the standard link state advertisement header. The type field contains a 1, and the link state ID field contains the switch ID of the advertising switch.

links

This 2-octet field contains the number of links described by this advertisement. This value must be equal to the total number of functioning network links attached to the switch.

Link ID

This 10-octet field identifies the other switch that originates link state advertisements for the link, providing a key for accessing other link state advertisements for the link. The value here is based on the link type, as follows:

- For point-to-point links, this field contains the switch ID of the neighbor switch connected to the other end of the link.
- For multi-access links, this field contains the switch ID of the designated switch for the link.

Link data

This 10-octet field contains additional data necessary to calculate the set of best paths. Typically, this field contains the interface ID of the link.

Link type

This 1-octet field contains the type of link being described. Possible values are as follows:

1 Point-to-point link
2 Multi-access link

TOS

This 1-octet field contains the number of nonzero type of service metrics specified for the link. Since the current version of VLSP does not support routing based on nonzero types of service, this field contains a value of zero.

TOS 0 metric

This 2-octet field contains the cost of using this link for the zero TOS. This value is expressed in the link state metric and must be greater than zero.

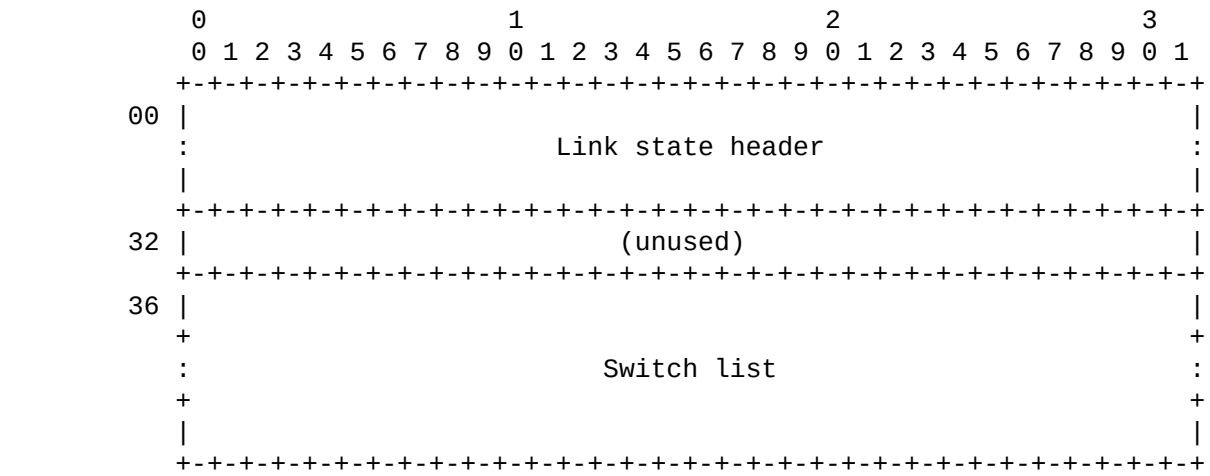
Note that the last five fields are repeated for all functioning network links attached to the advertising switch. If the interface state of attached link changes, the switch must originate a new instance of the switch link advertisement.

11.3 Network Link Advertisements

A network link advertisement is originated by the designated switch of each multi-access network link. The advertisement describes all switches attached to the link that are currently fully adjacent to the designated switch, including the designated switch itself. See Section 8.1 for details on originating a switch link advertisement.

Network link advertisements are not generated for point-to-point network links.

The format of a network link advertisement is show below.



Link state header

This 32-octet field contains the standard link state advertisement header. The type field contains a 2, and the link state ID field contains the switch ID of the designated switch.

Switch list

The switch IDs of all switches attached to the network link that are currently fully adjacent to the designated switch. The designated switch includes itself in this list.

12. Protocol Parameters

This section contains a compendium of the parameters used in the VLS protocol.

12.1 Architectural Constants

Several VLS protocol parameters have fixed architectural values. The name of each architectural constant follows, together with its value and a short description of its function.

AllSPFSwitches

The multicast switch ID to which Hello packets and certain other protocol packets are addressed, as specified in the destination switch ID field of the network layer address information (see Section 10.3). The value of AllSPFSwitches is E0-00-00-05-00-00-00-00.

AllDSwitches

The multicast switch ID to which Link State Update packets and Link State Acknowledgment packets are addressed, as specified in the destination switch ID field of the network layer address information (see Section 10.3), when they are destined for the designated switch or the backup designated switch of a network link. The value of AllDSwitches is E0-00-00-06-00-00-00-00.

LSRefreshTime

The interval at which the set of best paths recalculated if no other state changes have forced a recalculation. The value of LSRefreshTime is set to 1800 seconds (30 minutes).

MinLSInterval

The minimum time between distinct originations of any particular link state advertisement. The value of MinLSInterval is set to 5 seconds.

MaxAge

The maximum age that a link state advertisement can attain. When an advertisement's age reaches MaxAge, it is redistributed throughout the switch fabric. When the originating switch receives an acknowledgment for the advertisement, indicating that the advertisement has been removed from all neighbor Link state retransmission lists, the advertisement is removed from the originating switch's database. Advertisements having age MaxAge are not used to calculate the set of best paths. The value of MaxAge must be greater than LSRefreshTime. The value of MaxAge is set to 3600 seconds (1 hour).

MaxAgeDiff

The maximum time disparity in ages that can occur for a single link state instance as it is distributed throughout the switch fabric. Most of this time is accounted for by the time the advertisement sits on switch output queues (and therefore not aging) during the distribution process. The value of MaxAgeDiff is set to 900 seconds (15 minutes).

LSInfinity

The link state metric value indicating that the destination is unreachable. It is defined to be a binary value of all ones.

12.2 Configurable Parameters

Many of the switch interface parameters used by VLSP may be made configurable if the implementer so desires. These parameters are listed below. Sample default values are given for some of the parameters.

Note that some of these parameters specify properties of the individual interfaces and their attached network links. These parameters must be consistent across all the switches attached to that link.

Interface output cost(s)

The cost of sending a packet over the interface, expressed in the link state metric. This is advertised as the link cost for this interface in the switch's switch link advertisement. The interface output cost must always be greater than zero.

RxmtInterval

The number of seconds between link state advertisement retransmissions for adjacencies established on this interface. This value is also used when retransmitting Database Description packets and Link State Request packets. This value must be greater than the expected round-trip delay between any two switches on the attached link. However, the value should be conservative or needless retransmissions will result. A typical value for a local area network would be 5 seconds.

InfTransDelay

The estimated number of seconds it takes to transmit a Link State Update packet over this interface. Link state advertisements contained in the Link State Update packet must have their age incremented by this amount before transmission. This value must take into account the transmission and propagation delays for the interface and must be greater than zero. A typical value for a local area network would be 1 second.

Switch priority

An 8-bit unsigned integer. When two switches attached to the same network link contend for selection as the designated switch, the switch with the highest priority takes precedence. If both switches have the same priority, the switch with the highest base MAC address becomes the designated switch. A switch whose switch priority is set to zero is ineligible to become the designated switch on the attached link.

HelloInterval

The length of time, in seconds, between the Hello packets that the switch sends over the interface. This value is advertised in the switch's Hello packets. It must be the same for all switches attached to a common network link. The smaller this value is set, the faster topological changes will be detected. However, a smaller interval will also generate more routing traffic. A typical value for a local area network would be 10 seconds.

SwitchDeadInterval

The length of time, in seconds, that neighboring switches will wait before declaring the interface down once they stop receiving Hello packets over the interface. This value is advertised in the switch's Hello packets. It must be the same for all switches attached to a common network link and should be some multiple of the HelloInterval parameter. A typical value would be 4 times HelloInterval.

13. End Notes

[1] During calculation of the set of best paths, a network link advertisement must be located based solely on its link state ID. Note, however, that the lookup in this case is still well defined, since no two network advertisements can have the same link state ID.

[2] It is instructive to see what happens when the designated switch for a network link fails. Call the designated switch for the link S1 and the backup designated switch S2. If switch S1 fails (or its interface to the link goes down), the other switches on the link will detect S1's absence within SwitchDeadInterval seconds. All switches may not detect this condition at precisely the same time. The switches that detect S1's absence before S2 does will temporarily select S2 as both designated switch and backup designated switch. When S2 detects that S1 is down, it will move itself to designated switch. At this time, the remaining switch with the highest switch priority will be selected as the backup designated switch.

[3] Note that it is possible for a switch to resynchronize any of its fully established adjacencies by setting the neighbor state back to ExStart. This causes the switch on the other end of the adjacency to process a SeqNumberMismatch event and also revert to the ExStart state.

[4] When two advertisements have different checksum values, they are assumed to be separate instances. This can occur when a switch restarts and loses track of its previous sequence number. In this case, since the two advertisements have the same sequence number, it is not possible to determine which advertisement is actually newer. If the wrong advertisement is accepted as newer, the originating switch will originate another instance.

[5] An instance of an advertisement is originated with an age of MaxAge only when it is to be flushed from the database. This is done either when the advertisement has naturally aged to MaxAge, or (more typically) when the sequence number must wrap. Therefore, a received instance with an age of MaxAge must be processed as the most recent in order to flush it properly from the database.

[6] MaxAgeDiff is an architectural constant that defines the maximum disparity in ages, in seconds, that can occur for a single link state instance as it is distributed throughout the switch fabric. If two advertisements differ by more than this amount, they are assumed to be different instances of the same advertisement. This can occur when a switch restarts and loses track of its previous sequence number.

[7] This is how the link state request list is emptied, causing the neighbor state to change to Full.

14. Security Considerations

Security concerns are not addressed in this document.

15. References

[Perlman] Perlman, R., Interconnections: Bridges and Routers. Addison-Wesley Publishing Company. 1992.

[RFC905] McKenzie, A., "ISO Transport Protocol specification ISO DP 8073", RFC 905, April 1984.

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

[RFC1700] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994.

[IDsfvlan] Ruffen, D., Len, T. and J. Yanacek, "Cabletron's SecureFast VLAN Operational Model", RFC 2643, August 1999.

[IDhello] Hamilton, D. and D. Ruffen, "Cabletron's VlanHello Protocol Specification", RFC 2641, August 1999.

16. Author's Address

Laura Kane
Cabletron Systems, Inc.
Post Office Box 5005
Rochester, NH 03866-5005

Phone:(603) 332-9400
EMail: lkane@ctron.com

17. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.