IS-IS Protocol Hardware Implementation for VPN Solutions

MOHAMED ABOU-GABAL, RAYMOND PETERKIN, DAN IONESCU
School of Information Technology and Engineering (SITE)
University of Ottawa
161 Louis Pasteur, P.O. Box 450, Station A, Ottawa, Ontario, K1N 6N5
CANADA

Abstract: - One of the major problems related to the growth of network technology is the limitations it presents when the protection and the resilience of a network has to be assured for fast recovery as is the case for 1 and 10 Gbps links. It is well known that calculating the shortest path requires significant computational time. Stream-based applications (such as voice and video) running on Gigabit links can suffer dramatic packet loss during link failures, introducing large delays on receiving hosts in the process. In order to alleviate the delay in such networks, this paper introduces hardware architecture for the IS-IS protocol. The architecture implements mathematical algorithms for calculating the shortest path in FPGAs. Such an implementation improves the shortest path computation time, allowing for a rapid calculation of the second path.

Key-Words: - IS-IS, hardware, FPGA, VPN, network, OSPF, Dijkstra

1 Introduction

In order to overcome and engineer some of the performance issues of packet based networks, hierarchical and logical routing architecture protocols were developed in the area of packet networks. Some of these protocols include Hierarchical Peer Network-to-Network interface (HPNNI) [1] which is applied to ATM networks, Open Shortest Path First (OSPF) [2], Intermediate-System to Intermediate-System (IS-IS) [3] and Border Gateway Protocol (BGP) [4] which are used to route traffic in autonomous IP networks. Most of these protocols are based on computing Dijkstra's, Bell-man Ford's or other algorithms to find the shortest path from a source node to a destination node.

OSPF and IS-IS protocols are well known protocols for supporting routing in a network layer protocol with datagram service. While both protocols have numerous similarities (authentication, flooding mechanisms, small control packets, etc.), there are differences that become significant. In the construction, transmission and processing of link state PDUs (LSPs), OSPF is memory intensive as more bits are transmitted with larger link state headers and larger link state packets [5]. Consequently, OSPF requires more bandwidth than IS-IS to transmit information to its neighbors. However, IS-IS packets cannot be easily differentiated at lower (kernel) levels (because IS-IS is transmitted on top of the data link layer) and most IS-IS fields are of variable length, making packet parsing relatively difficult. Therefore, the effective trade off between OSPF and IS-IS is that OSPF has optimized processing with increased packet sizes while IS-IS has minimized storage and bandwidth requirements with slow processing time.

IS-IS outperforms OSPF in link state packet arrival times which is considered the most important performance criterion in analyzing how routing information is transmitted throughout networks [6]. Therefore, IS-IS is the preferred protocol for disseminating routing information over broadcast networks. For these reasons, we pursue the development of hardware architecture for IS-IS.

Hardware architectures for IS-IS or routing protocols in general can be used to optimize processing tasks that otherwise serve as significant bottlenecks in computer communications. The architecture in [7] illustrates how the shortest path can be calculated in hardware using Dijkstra's algorithm. That shortest path architecture can be used with any protocol requiring shortest path calculations to optimize routing computations. An IS-IS protocol architecture is given in [8] illustrating several modules interacting with each other to perform IS-IS tasks, including a separate module to calculate the shortest path. A similar architecture could be implemented in hardware with FPGAs to optimize processing, addressing the advantage of OSPF compared to IS-IS, while maintaining low storage and bandwidth requirements.

This paper will present a new hardware architecture for the IS-IS protocol. This architecture allows for a network topology to be directly entered or removed as IS-IS performs pertinent routing operations. The paper is organized in the following manner. Section 2 provides a brief overview of the IS-IS protocol. Section 3 mathematically describes the shortest path problem while section 4 illustrates a linear programming solution specific to shortest path routing. Section 5 describes why hardware should be used to address traffic shaping in routing protocols. Section 6 describes the hardware architecture of IS-IS as it would be implemented on an FPGA. Section 7 describes experimental results achieved thus far with respect to calculating the shortest path for IS-IS. Finally, in Section 8 the conclusion to this work is listed, summarizing relevant aspects of the architecture and describing future work.

2 IS-IS Overview

IS-IS is an interior gateway protocol and link state protocol which uses Dijkstra's algorithm to find the shortest path between two routers within an area. In this section of the paper a brief overview of IS-IS is given by describing the network elements and operations.

Fig.1 shows the network elements involved in an IS-IS domain. A host is referred to as an End System (ES), a router is called an Intermediate System (IS), and a designated router is referred to as a Designated Intermediate System (DIS). The data link interface address (MAC address) is known as a Sub-network Point of Attachment (SNPA). The network layer address (IP Address) is known as a Network Service Access Point (NSAP). IS-IS has two levels of area routing; level 1 (L1) and level 2 (L2). In L1, ISs route within their own areas. If the destination is not within the area then data will be routed to the closest L2 IS. L2 acts as a backbone where ISs route towards other areas. Certain ISs can be configured for both L1 and L2. Routing information packets used among ISs are known as Link State PDUs (LSPs). Information collected by LSPs is stored in a Link State Database (LSDB). If an IS belongs two levels (L1L2) then that IS will have two LSDBs; one for each level.

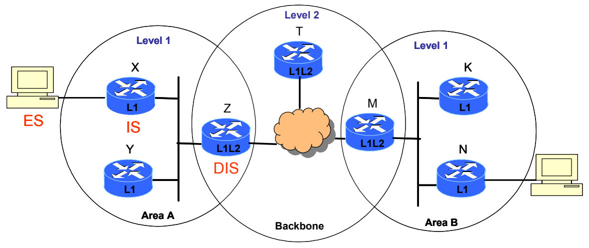

Fig.1. IS-IS Domain

Fig.2 illustrates a high level operation of IS-IS. The hello state illustrates the activities performed in establishing a connection between two ISs. From Fig.1, it is assumed that Y is an IS coming up. Z acts as a DIS (i.e. performs the flooding of LSPs) and as a Pseudo-node (PSN) (i.e. the virtual node that emulates a broadcast link). When Y comes up it will be idle and it must establish its adjacencies. Y will send out IS-IS Hello (IIH) packets with a null SNPA. Z receives IIH packets and checks its LSDB to determine if an adjacency already exists. The IIH packet is ignored if the adjacency exists; otherwise Z creates a new adjacency and sends an IIH packet to Y. Y receives the IIH packet from Z and returns an acknowledgement. Z returns an additional IIH packet to completely establish the connection.

The next state is the Database (DB) Synchronization state where exchanges of LSPs occur. The sequence number integrated with LSPs is used to differentiate older LSPs from newer LSPs. From Fig.2, after Y has established its adjacency, it will send out its LSP with sequence number 1. Upon receiving the LSP Z sends other LSPs to Y. Y uses these LSPs to create its L1 LSDB.

Once databases are synchronized, all ISs compute the shortest path within their area in a state known as Shortest Path First (SPF).

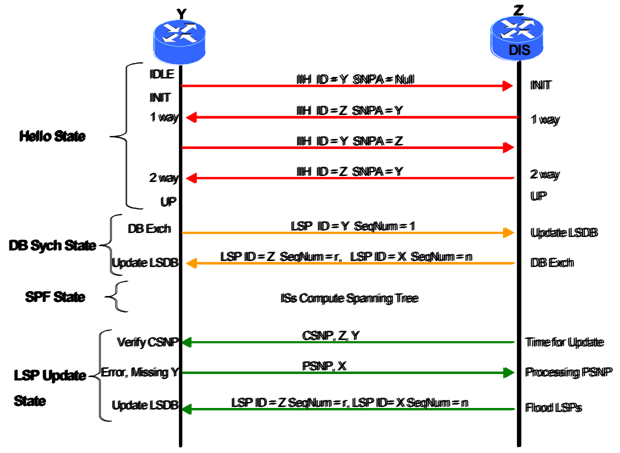

Fig.2. IS-IS Operations

The next state is known as LSP Update. In this state two kinds of sequence numbers are introduced. The first kind is the Complete Sequence Number PDU (CSNP) and the second kind is the Partial Sequence Number PDU (PSNP). CSNPs are used to determine if ISs have the current copies of all LSPs. PSNPs are used by ISs to respond to CSNP if the CSNP had a missing LSP. Fig.2 shows an example of this scenario. After a specific period of time, Z advertises CSNPs with summaries of all known LSPs. Y discovers that the X LSP is missing. Hence, Y sends a PSNP request for the missing LSPs. Z processes the PSNP and floods LSPs that includes the missing X. Y receives the complete LSPs, updates its LSDB and recomputes the shortest path.

3 Shortest Path Problem

The following notations can be used to describe the shortest path problem; "G" denotes a graph, "N" denotes a set of nodes, "A" represents a set of arcs, "s" symbolizes the source node and "t" denotes the termination node.

$$G = (N,A) \tag{1}$$

$$\{x_{ij} \mid (i,j) \in A\} \tag{2}$$

where x_{ij} denotes the arc/link connecting the node i to node j. The following VPN specific network from [9] expands this definition to include customer nodes (c), customer edges (ce), provider nodes (pe) and provider edges (p).

$$N = N_{ce} \cup N_{pe} \cup N_p \cup N_a \tag{3}$$

$$N_{ce} \cap N_{pe} = \varnothing \tag{4}$$

$$N_{pe} \cap N_p = \varnothing \tag{5}$$

$$N_{ce} \cap N_p = \varnothing \tag{6}$$

With this notation, the shortest path problem is described as the minimization of the following equation

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \tag{7}$$

where "c_{ij}" denotes the link cost between node i and node j. The equation is minimized subject to the following constraints.

$$\sum_{\{j|(i,j)\in A\}} x_{ij} - \sum_{\{j|(j,i)\in A\}} x_{ij} = \begin{cases} 1, i = s \\ -1, i = t \\ 0, otherwise \end{cases} \tag{8}$$

$$0 \le x_{ij}, \forall (i,j) \in A \tag{9}$$

Solving the shortest path problem in software typically requires O(n^2) or O(nlogn) algorithms. Those algorithms use data structures whose operations run in O(1) to O(n) time. Significant inefficiency can exist depending how which algorithms and data structures are used to solve the shortest path problem. The use of reconfigurable hardware for the shortest path problem helps to alleviate the inefficiency observed in software.

4 Shortest Path Routing

Optimal performance in terms of bandwidth consumed on each link can be achieved with a set of shortest paths by solving the following linear program and dual shown in [10]. The linear program can be described as

$$\min \sum_{(i,j)\in A} \sum_{r\in R} d_r X_{ij}^r$$

$$\begin{aligned} &\text{subject to} \\ &\sum_{j:(j,i)\in A} X_{ij}^r - \sum_{j:(i,j)\in A} X_{ij}^r = 0, i \ne s_r, t_r \; r \in R \\ &\sum_{j:(j,i)\in A} X_{ji}^r - \sum_{j:(i,j)\in A} X_{ij}^r = 1, i = t_r \; r \in R \\ &\sum_{r\in R} d_r X_{ij}^r \le c_{ij} \; (i,j) \in A \\ &0 \le X_{ij}^r \le 1 (i,j) \in A, r \in R \end{aligned} \tag{10}$$

where T(s_r, t_r) = d_r for assumed traffic matrix T, source (ingress) node s and destination (egress) node t. X_{ij}^r is the fraction of traffic for r through link (i,j). The dual to the linear program is described as

$$\max \sum_{r\in R, t\in V} U^r_{t_r} - \sum_{(i,j)\in A} c_{ij} W_{ij}$$

subject to

$$U^r_j - U^r_i \leq W_{ij} + 1 \qquad \forall r \in R, (i,j) \in A \tag{11}$$

$$W_{ij} \geq 0$$

$$U^r_{s_r} = 0.$$

where W_{ij} denotes the link weight for link (i,j). The linear program and dual shown here illustrate the importance of efficient processing for shortest path calculations, as a different set of next hops may be generated source-destination pair where traffic may be forwarded. The architecture in this paper is also linear so efficiency in shortest path routing is of significant importance. The following section illustrates the justification of FPGAs in performing traffic shaping tasks through routing protocols

5 Traffic Shaping and Hardware

IS-IS and other routing protocols play substantial roles in traffic shaping (altering traffic to increase network efficiency). The potential improvements offered by reconfigurable hardware (FPGAs, Systems-on-Chip) to traffic shaping and other computer communications tasks is best illustrated by examining performance requirements.

The most restrictive traffic shaping requirements come from high speed devices that must operate at high rates. The information given in [11] illustrates that slot durations range from 2us to 42us for connections of at least 155Mbps where the minimum packet sizes were 40 bytes to 64 bytes. Traffic shaping at those rates is only achievable with dedicated hardware. These rates are of particular importance in IP networks since packets operations are expensive due to variable packet size. Furthermore FPGAs are preferable as hardware implementation devices because they are reconfigurable, so a given platform can be optimized as much as possible.

6 Hardware Architecture

In this section of the paper the IS-IS hardware architecture will be presented as follows; a description of the inputs and outputs of the IS-IS system and a system description is given first. Then more detailed discussions are given for the control unit state machine and the data path. Fig.3 shows the inputs and outputs of the IS-IS system. The IS-IS system takes the following inputs:

- *ingressPacket,* which is where incoming packets will be received.
- *afiValue,* which will be used to set the OSI NSAP address (the first octet), the value of this field indicates the top-level addressing domain associated with the NSAP.
- *areaAddress,* which along with *afiValue* form what's called the areaID.
- systemID, which is the MAC address of the IS.
- *nsel,* which specifies a user of the network layer service, if left un-initialized it will be set to zero.
- *psnID,* the ID that will be used by the DIS to act as a PSN.
- *dis,* which acts as the DIS.
- *clock,* this signal provides a clocking source to the system for synchronization.
- *reset,* this signal is used to reset IS-IS system to its default configuration.

The IS-IS system provides the following outputs:

- *egressPacket,* is the interface at which IS-IS system will be outputting packets such as LSPs, CSNP, PSNP,etc.
- *isState,* informs the IS-IS system user about the state of the IS.

The IS-IS system is composed of two major subsystems, a control unit and a data path (as shown in Fig.4). The control unit is composed of 4 processors, the Main Processor (MP) is responsible for invoking and updating the other 3 processors based on feedback from all other processors. The

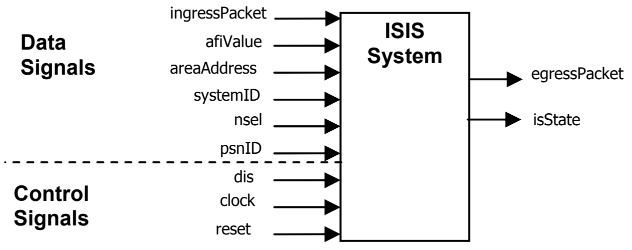

Fig.3. IS-IS I/O

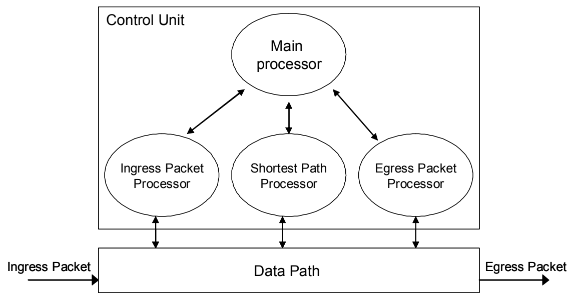

Fig.4. IS-IS system view

ingress packet processor (IPP) is responsible for processing and buffering packets received on the ingress path. It also updates the MP with status signals on the type of packets received. The Shortest Path Processor (SPP) is responsible for computing the SPF. The Egress Packet Processor (EPP) handles all tasks relating to sending packets as the result of LSP advertisements, responding or acknowledging packets.

6.1 Control Unit

The control unit is presented in the state machine shown in Fig.5. The Hello state is where the adjacency establishment is achieved. If an adjacency was already established then the control unit ignores the hello packet and goes back to the IDLE state. If an adjacency was not established then the control unit remains in the hello state and performs all necessary computations. In the DB Exchange state the control unit receives and advertises LSPs to make sure that its LSDBs are synchronized with the latest network topology. Upon completing the DB Exchange state, the control unit processes buffered LSPs and stores them into the appropriate LSDB, in a state known as LSDB Update. Once the LSDB update state is

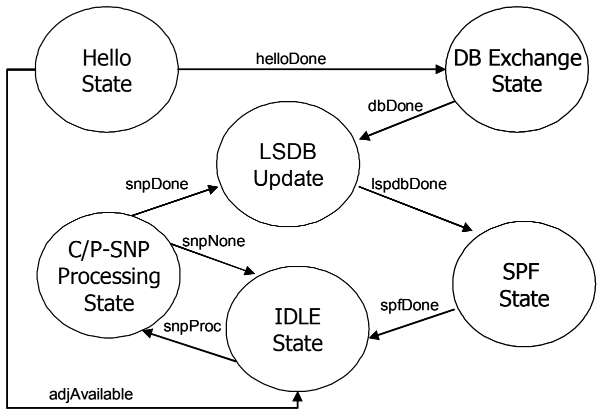

Fig.5. IS-IS main control unit

complete, the SPF state is executed and the shortest path is computed. The control unit moves to an IDLE state where it waits for timers to be triggered. In the case of a DIS deployment, the triggered timers force the control unit to enter the C/P-SNP Processing state where CSNP packets are sent out to the network. The control unit stays in the C/P-SNP Processing state for a certain amount of time to ensure that no PSNPs are received. In a regular IS deployment, the control unit moves to the C/P-SNP Processing state if a CSNP is received on the ingress path. In some cases, if a CSNP was missing some LSP summaries, then LSP advertisements will be flooded on the network and as a result the control unit will go to the LSDB Update state and SPF states.

6.2 Data Path

As shown in Fig.6, the data path is composed of ingress components, egress components, LSDBs, system inputs, the SPP data path and the muxDemux components. The ingress and egress components are buffers that store packets. The ingress components are denoted by "in" in front of the component name and the egress components are denoted by "eg". Below is brief description of each buffer:

- IIH, buffers hello packets.
- DB, buffers database packets.
- LSP-L1, buffers Level 1 LSP packets.
- LSP-L2, buffers Level 2 LSP packets
- CSNP, buffers CSNP packets.
- PSNP, buffers PSNP packets.

There are two major LSDBs; the active LSDB-L1 and the active LSDB-L2. The active LSDB-L1 stores the processed LSPs for L1 and the active LSDB-L2

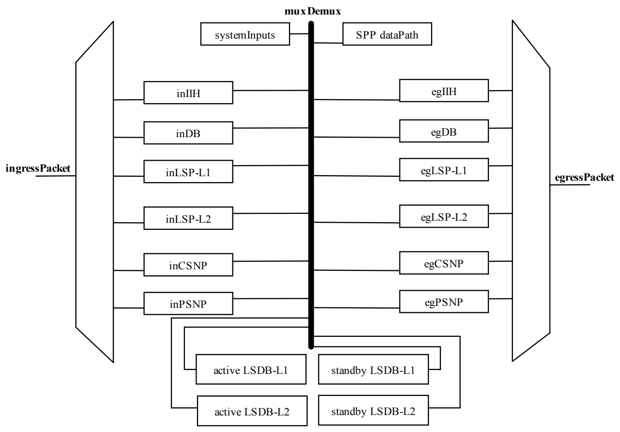

Fig.6. IS-IS data path major components

stores the processed LSPs for L2. The standby LSDBs are initialized to zero. It is used during the deletion (link failures or IS crashes) or the addition of LSPs. In the scenario of adding a new LSP, the IS-IS system adds the new LSP to the standby LSDB, copies the active LSDB to the standby LSDB, then it activates the standby LSDB and deactivates the active LSDB (the standby LSDB becomes active and the active LSDB becomes standby). In the scenario of deleting a LSP, the modification (replacing LSPs with zeros) is done on the active LSDB, then the IS-IS system copies the active LSDB (skipping previously inserted zeros) to the standby LSDB, then it activates the standby LSDB and deactivates the active LSDB. The switching between active and standby LSDBs occurs whenever necessary. The system inputs component are registers where the IS-IS inputs are stored. The SPP data path described in [7] is used here. The muxDemux component represents multiplexers and demultiplexers (for regulating data flow) that were hidden from Fig.6 to simplify the data path diagram.

7 Experimental Results

Simulation results have been accumulated thus far with respect to calculating the shortest path for IS-IS. Other aspects of IS-IS hardware implementation are not completed and consequently no results can be given at this time. Networks of seven nodes or less were used to generate results. Hardware simulations were performed using Cadence software while software benchmarks were found with Java implementations of Dijkstra's algorithm.

The shortest path was calculated in approximately 160 – 200 clock cycles in hardware simulations. In software, those networks had their shortest paths determined in about 0.2 ms on a 267 MHz with a standard deviation of approximately 8% over 50 samples per network. Those results indicate that the shortest path can be calculated in approximately 200 cycles in hardware and the equivalent network has its shortest path calculated in 57000 cycles in a typical software environment.

8 Conclusion

The IS-IS hardware architecture presented in this paper is flexible in adapting to network topology changes. The architecture is scalable through FPGA implementation. Delays for stream-based applications can be minimized by using this architecture as demonstrated in the experimental results achieved for computing the shortest path.

References:

[1] ATM Forum Technical Committee, *Private Network-Network Interface Specification*, March 1996.

[2] J.Moy, *OSPF Anatomy of an Internet Routing Protocol*, Addison-Wesley, February 1998.

[3] White, Russ, *IS-IS: Deployment in IP Networks*, Addison-Wesley, 2003.

[4] Y. Rekhter, P. Gross, *Application of the Border Gateway Protocol in the internet*, RFC 1772, March 1995.

[5] Perlman, Radia, A Comparison Between Two Routing Protocols: OSPF and IS-IS, *IEEE Network Magazine*, Vol.5, No.5, 1991, pp. 18 – 22.

[6] Sharon, Oran, Dissemination of Routing Information in Broadcast Networks: OSPF versus IS-IS, *IEEE Network*, Vol.15, No.1, 2001, pp. 64.

[7] Abou-Gabal, Mohamed, Peterkin, Raymond, Ionescu, Dan, Lambiri, Cristian, Groza, Voicu, A Shortest Path Processor for Traffic Engineering of VPN Services, *PERIODICA POLITECHNICA, Transactions on AUTOMATIC CONTROL and COMPUTER SCIENCE*, Vol.49, No.63, 2004.

[8] Yang, Mijeong, Hahm, Jinho, Kim, Youngsun, and Kim, Sangha, Design and Implementation of the IS-IS Routing Protocol with Traffic Engineering, *APCC 2003, the 9th Asia-Pacific Conference on Communications*, Vol.3, 2003, pp. 1103.

[9] Lambiri, Cristian, Ionescu, Dan, Ionescu, Bogdan. Service Control in Connection Oriented Networks, *Recent Advances in Communications and Computer Science (WSEAS)*, 2003, pp. 266.

[10] Sridharan, Ashwin, Guerin, Roch, Diot, Christophe, Achieving Near-Optimal Traffic Engineering Solutions for Current OSPF/IS-IS Networks, *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol.3, 2003, pp. 1168.

[11] Orphanoudakis, T, Leligou, H.-C., Kornaros, George, Charopoulos, Ch. Accurate Scheduler Implementation for Shaping Flows of Variable Length Packets in High-Speed Networking Applications, *Recent Advances in Communications and Computer Science (WSEAS)*, 2003, pp. 96.