Interdomain Path Computation: Challenges and Solutions for Label Switched Networks

Faisal Aslam, Zartash Afzal Uzmi, and Adrian Farrel

ABSTRACT

For label switched networks, such as MPLS and GMPLS, most existing traffic engineering solutions work in a single routing domain. These solutions do not work when a route from the ingress node to the egress node leaves the routing area or autonomous system of the ingress node. In such cases, the path computation problem becomes complicated because of the unavailability of the complete routing information throughout the network. This is because service providers usually choose not to leak routing information beyond the routing area or AS for scalability constraints and confidentiality concerns. This article serves two purposes. First, it provides a description of the existing and ongoing work in interdomain TE within the IETF. This information is currently found in various Internet drafts and has not yet been collectively presented in a single document. To this end, a summary of both existing path computation architectures — *PCE-based* and *per-domain* — is provided. Second, it compares two per-domain path computation schemes in terms of the total number of LSPs successfully placed and average number of domains crossed, without assuming availability of complete topology information. We notice that the two per-domain path computation schemes, proposed in [1, 2], have comparable path computation complexities and setup latencies.

This work was supported in part by Cisco Systems under their URP program.

[1] *The term* routing domain*, as used in this article, refers to a network under a single administration with common policies. A routing domain, therefore, may be an autonomous system or a routing area within the autonomous system.*

INTRODUCTION

The emergence of mission-critical and other multimedia applications such as voice over IP (VoIP), videoconferencing, e-commerce, and virtual private networks (VPNs) has translated into stringent real-time quality of service (QoS) requirements for carrier networks. A key factor in meeting the QoS requirements of such applications is the ability to route traffic along explicit paths computed through constraint-based routing. The destination-based shortest path routing paradigm employed in IP routing does not support routing network traffic along explicit paths. However, the emergence of label switching paradigms such as multiprotocol label switching (MPLS) and generalized MPLS (GMPLS) has overcome this limitation by presenting the ability to establish a label switched path (LSP) between two points in an IP network. This ability to do traffic engineering (TE) using MPLS maintains the flexibility and simplicity of an IP network while exploiting the advantages of an asynchronous transfer mode (ATM)-like connection-oriented network.

Ingress routers of an MPLS network classify packets into forwarding equivalence classes (FECs) and encapsulate them with labels before forwarding them along precomputed paths. The path a packet takes as a result of a series of label switching operations in an MPLS network is called a label switched path (LSP). For providing connectivity from ingress to egress, an LSP traverses a sequence of links that could be either physical links between adjacent nodes or logical links between two nodes that may or may not be adjacent.

LSPs may be computed by using a constrained shortest path first (CSPF) algorithm that essentially finds a shortest path between two network nodes subject to constraints such as maximum delay, minimum available bandwidth, and resource class affinity. Thus, the constraints dictate how the traffic should be engineered through the network; for this reason, the paths computed under the constraints are called traffic engineered (TE) paths. For the computation of a TE path an LSP would traverse, a CSPF algorithm uses TE information, such as the remaining or reserved bandwidth along a logical or physical link, advertised throughout the network. Resources along a computed TE path are reserved during label distribution, using protocols such as Constraint-Based Routing Label Distribution Protocol (CR-LDP) and Resource Reservation Protocol with TE (RSVP-TE) [3].

Existing solutions for traffic engineering in MPLS and GMPLS networks are mostly limited to work within a single routing domain[1] and do not work when traffic leaves the boundaries of the routing area or autonomous system (AS) of an LSP ingress node [4]. This is primarily because in such a case, the ingress node has limited visibility of the topology and TE resource information (e.g., bandwidth, delay) outside its own routing domain, and therefore cannot compute an end-to-end optimal path that spans multiple domains. The definition of optimal path may vary depending on the goals; however, in the literature, an optimal interdomain path refers to a constrained shortest path that would be computed if there was just a single routing domain, and complete topological and resource information were available during computation [5]. In addition, one may impose an extra constraint of minimizing the number of domains traversed by the computed route [5].

In this article we provide a summary and comparison of various interdomain path computation schemes. The primary goal of all these schemes is to compute an optimal or near-optimal path while assuming availability of limited topological information and fulfilling most interdomain path computation requirements. Such requirements may include:
- No advertisement of internal topology or resource information outside domain boundaries. This is critical for security, confidentiality, and preservation of the scalability of the routing protocol used within a routing area (the interior gateway protocol, IGP) and the routing protocol that exchanges information between the networks (the border gateway protocol, BGP).
- No unreasonable increase in IGP load and preservation of RSVP-TE scalability [4].

The rest of the article is organized as follows. We provide relevant background material. Existing interdomain path computation schemes are given while we detail our evaluation methodology. Simulation setup and results comparing two per-domain schemes are discussed, and we then conclude the article.

Path Setup Mechanisms

In a network that employs TE, destination-based hop-by-hop forwarding may be replaced by a mechanism in which traffic is sent along explicit paths from the ingress node to the egress node. There are two distinct ways an interdomain explicit path can be computed for subsequent setup:
- An initial path is computed by the ingress node, and refined by intermediate nodes along the path to the egress node without any help from a central entity. Due to confidentiality constraints across routing domains, an ingress node does not usually have the capability to compute a hop-by-hop path to the egress. Instead, a path that includes some, but not all, intermediate nodes to the egress is computed, and an LSP setup request is forwarded along this route also referred to as a loose route. The loose route gets refined as the LSP setup request progresses along the path.
- The second way of computing an interdomain path is by using a path computation element (PCE). In this case, too, the PCE may not have visibility of the complete network map and is capable of computing only a loose route from an ingress to an egress node. The computed loose route is returned by the PCE in an explicit route object (ERO) for use in signaling, and the loose route is refined as the LSP setup request progresses along the path.

Existing solutions for traffic engineering in MPLS and GMPLS networks are mostly limited to work within a single routing domain and do not work when traffic leaves the boundaries of the routing area or autonomous system (AS) of an LSP ingress node.

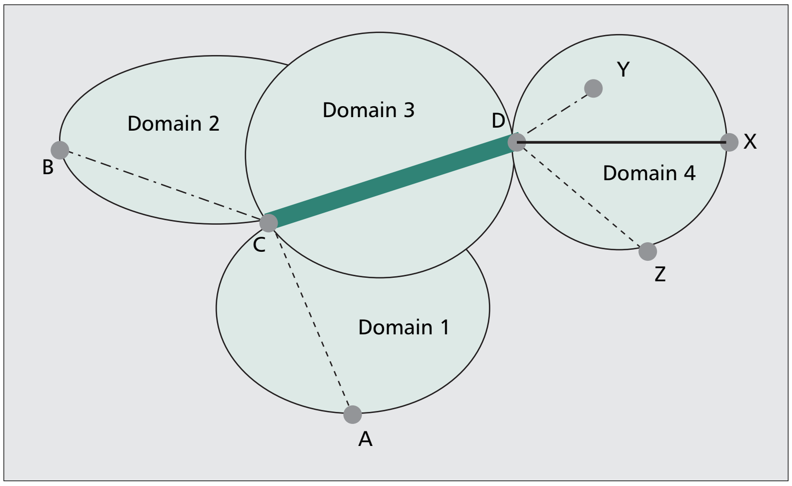


Figure 1. *LSP hierarchy or nesting. Three interdomain LSPs are created: from node A to node Z, from node B to node Y, and from node C to node X. A single intra-domain H-LSP accommodates these three inter-domain LSPs from node C to node D.*

After the computation of a loose interdomain route, whether the computation is carried out at the ingress or at a PCE, an ERO is created and signaled. An intermediate node may have to expand the ERO, received in an RSVP-TE *Path* message, if the loose hop does not provide sufficient information to determine the next hop along the path to the egress. An interdomain LSP can be set up using four different methods:
- Contiguous LSPs
- Nesting or LSP hierarchy
- LSP stitching
- Hybrid methods

The choice of which option to use may depend on several factors including the needs of a particular path computation technique, the requirements of the application using LSPs, service provider policies, network topology, and the switching capabilities of the network [6]. The four methods of interdomain LSP setup are summarized below.

Contiguous LSP

A contiguous LSP is a single end-to-end LSP established across multiple domains. The actual setup requires RSVP-TE signaling procedures described in [3]. A contiguous LSP is established without any interaction, such as aggregation, with any other LSP. For this rea-

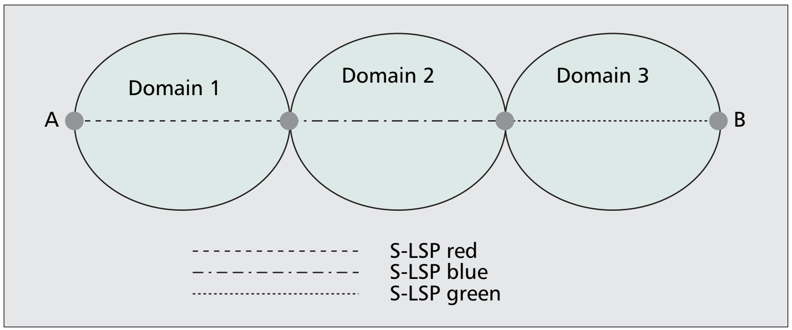


Figure 2. *Depicting LSP stitching. An interdomain LSP from ingress node A to egress node B is created using three intradomain S-LSPs.*

son, the same identifier for the LSP (the session ID) is used on each hop of the LSP across all domains.

LSP NESTING

LSP hierarchy (also known as nesting [7]) allows one or more LSPs, along a common part of their path, to be carried within a single hierarchical-LSP (H-LSP). A pre-established H-LSP traversing many links may be advertised as a single link. A link, physical or logical, whose TE information is advertised to help CSPF compute a path is referred to as a TE link. When nodes advertise an H-LSP as a TE link, other nodes can include this TE link in their path computations, as they would for any other link, without knowledge of the existence of the underlying H-LSP. An LSP using one or more H-LSPs appears as a single end-to-end LSP in the data plane, as shown in Fig. 1. As this figure indicates, overlapping portions of many existing LSPs can be nested together in the data plane so that a single end-to-end LSP is established for such overlapping portions.

LSP STITCHING

The concept of LSP stitching [8] is similar to nesting. However, unlike an H-LSP, in the case of stitching, a stitching LSP segment (S-LSP) can accommodate only one LSP. Using LSP stitching, many existing LSPs can be stitched together in the data plane so that a single end-to-end LSP is established, as indicated in Fig. 2.

LSP nesting and stitching allow an interdomain LSP be established using multiple intradomain H-LSPs or S-LSPs. The H-LSPs and S-LSPs are intradomain LSPs, and may either be pre-established or established dynamically upon the arrival of a TE LSP setup request.

HYBRID APPROACH

In the hybrid approach, an interdomain TE LSP may use contiguous, stitching, or hierarchical LSPs within different domains for an end-to-end path establishment. The selection of a particular method within a given domain depends on the LSP policy signaled by the ingress node and the domain's local policy. The path computation mechanisms are there to compute the path; hence, they do not depend on whether an LSP is established as a contiguous LSP or uses a stitching and nesting procedure within different domains.

INFORMATION SCENARIOS

The optimality of a computed constrained path and how loosely that path is specified depends on the amount of network information available at the ingress node. This leads to the definition of three information scenarios as given in [5]: multidomain visibility, partial visibility, and local domain visibility. These information scenarios impact the optimality of the computed path and are summarized below.

MULTIDOMAIN VISIBILITY

In this scenario the ingress node has sufficient information about the topology and TE resources of all relevant domains. Hence, the ingress node can compute a complete end-to-end interdomain optimal path. Thus, the ingress node can fully specify and signal an explicit end-to-end optimal path, in the ERO, from itself to the egress node. However, achieving this information scenario may not be practical given the requirement that a domain should never advertise TE resources and topological information outside its boundaries [4].

PARTIAL VISIBILITY

In this scenario the ingress node has full information about its own domain, and has information about the connectivity between domains as well as TE resources availability across other domains, but it does not have full visibility of the topologies *inside* other domains. Consequently, the ingress node is not able to provide, in the ERO, a fully specified strict explicit path from ingress node to the egress node. However, the ingress node can still supply some useful information about intermediate domains and signal this within the ERO. For example, the ingress node might supply an explicit path that comprises:

- Explicit hops from the ingress node to the local domain boundary
- Intermediate domains represented as *abstract* nodes or their entry points specified as loose hops, where an abstract node, despite appearing as a single node in the ERO, refers to a group of nodes, usually sharing a common prefix, whose internal topology is opaque to the ingress node of the LSP (e.g., a domain other than the domain of the ingress node) [3]
- A loose hop identifying the egress node

Note that it is possible that the ingress node does not have information about the TE resources of other domains, in which case the ingress node can try multiple paths one by one in order to successfully place the LSP along a path that satisfies the TE constraints.

LOCAL DOMAIN VISIBILITY

The ingress node has full visibility of its own domain and connectivity information only as far as determining an exit point from the current domain through one or more domain border routers (DBRs)—routers that connect two routing domains[2] — that may be suitable for carrying the LSP to its egress node. In this case the ingress node builds an explicit path that comprises just:

[2] *Examples of DBRs include IGP area border routers (ABRs) or AS border routers (ASBRs).*

- Explicit hops from the ingress node to the local domain boundary
- A loose hop identifying the egress node

Crankback signaling is a way using which a router indicates if it could not reserve resources along a TE path to the egress router.

Crankback Signaling

Crankback signaling facilitates rerouting of an LSP setup request around blocked or failed network elements in case a path setup attempt is unsuccessful [6]. In crankback signaling a router indicates if it could not reserve resources along a TE path to the egress router. This indication is sent to a router by a downstream router, which may try several alternate paths to the destination before signaling this information. Crankback signaling, which enhances existing RSVP-TE signaling to establish LSP tunnels [3], has additional applications including interdomain constrained path computation and restoration routing in the case of multiple network element failures.

In case of an unsuccessful path setup attempt (i.e., unsuccessful RSVP-TE *Path* message [3]), crankback information from a failed or blocked network element is propagated upstream using *PathErr* or *Notify* messages. This information includes the identity of the blocked or failed network element and the reason for path establishment failure (e.g., congestion, network element failure, or resource violation).

Upon receiving a *PathErr* message, an upstream routing point,[3] say RCOMP, saves the crankback information in its local crankback history table and discards the *PathErr* message. Subsequently, it attempts to reroute, avoiding the blocked elements specified in its crankback history-table. The routing point updates the local crankback history-table, with the latest crankback information after each failed path setup attempt, so that any successive rerouting attempts can avoid all known blocked network elements. A routing point may opt to discard the information stored in the local crankback history-table after an LSP is successfully established.

An RSVP *PathErr* message is generated by RCOMP if it is unable to successfully establish a path using any of its domain exit points. This *PathErr* message contains the summarized information from the history-table of RCOMP and is sent to the upstream nodes, such that an upstream routing point (further upstream from RCOMP) can benefit from the experiences of RCOMP. Furthermore, the upstream routing point should also avoid rerouting attempts through RCOMP because now it is established that no path exists through it.

The ingress node uses the LSP ATTRIBUTE object of the *Path* message to nominate the nodes that may perform the role of a routing point during the computation of a complete path. For this purpose, four rerouting flags are suggested [9]:

- *No Rerouting*: Only the ingress node may attempt to reroute; intermediate nodes may or may not supply crankback information.
- *End-to-End Rerouting*: Only the ingress node may attempt to reroute, while intermediate nodes must supply crankback information.
- *Boundary Rerouting*: Only intermediate DBRs or the ingress may attempt rerouting after receiving crankback information, while other nodes should supply crankback information.
- *Segment-Based Rerouting*: Any node may attempt rerouting after receiving crankback information. If a given node chooses not to perform rerouting then it should still supply crankback information upstream.

Existing Path Computation Schemes

Interdomain path computation schemes can be divided into two categories: PCE-based computation and per-domain path computation. Depending on the service provider requirements and functionality available on nodes, one may adopt either one of these techniques.

PCE-Based Path Computation

The PCE is an entity — a node or a process — responsible for computing an interdomain TE-LSP upon receiving a request from a path computation client (PCC), which could also be another process or a node. There could be zero, one, or more PCEs per domain; a PCE may or may not reside on the same node as its corresponding PCC. A path may be computed by either a single PCE node or a set of distributed PCE nodes that collaborate during path computation [10].

The PCE architecture mandates that a PCC should send a targeted path computation request to a particular PCE, using PCC-to-PCE communication, instead of broadcasting such a request to all the PCEs. Thus, PCCs maintain local information about the computation capabilities of various PCEs by either using static configuration or listening to the periodic advertisements generated by PCEs. This locally maintained information is used to select the appropriate PCE for a particular path computation job. In most cases the ingress node for each request serves as the PCC. To sum it up, when a new request arrives, the PCC first discovers a PCE using the PCE discovery mechanism [11], and then forwards the path computation request to the selected PCE using PCC-to-PCE communication [10].

A PCE may compute the end-to-end path itself if enough topology and TE information is available to it. Furthermore, the PCE architecture provides mechanisms for the resolution of path computation requests when an individual PCE does not have sufficient TE visibility. For example, a PCE may cooperate with other PCEs to determine intermediate loose hops, path segments that are kept confidential through the use of path keys, or even a full explicit path. This article restricts its consideration to per-domain path computation where the PCE in each domain is responsible for computing the next stage of the path toward the destination, and cooperating PCEs are left for future study. That is, we only consider the situation when nodes

[3] *The term routing point refers to a label switched router (LSR) that can perform path computation. There may be several routing points along the complete path from ingress to egress.*

A routing point RCOMP refines a loose route, received in an ERO, in the current domain such that all nodes within the current domain are identified. In doing so, RCOMP may notice that the ERO contains the domain exit point from the current domain.

along the path are responsible for path computation using their own PCE function or accessing local PCEs.

Per-Domain Path Computation

As discussed earlier, visibility of interdomain topology is practically limited at the ingress nodes; therefore, if cooperative PCE-based computation is not an option, the ingress node will only be able to determine a loose route to the egress node, unless the computed path does not need to leave the routing domain. This loose route must be refined in each domain through which it passes, by performing computations at a routing point within that domain. The refinement of a loose route within a domain simply means that all intermediate nodes within that domain are ascertained and included within the ERO. Such a path computation mechanism, in which nodes (routing points) along the loose route are responsible for path computation using their own PCE function or accessing PCEs within their own domain, is called per-domain path computation [1]. Thus, complete specification of all nodes from ingress to egress may require a series of per-domain path computations within each domain through which a loose route traverses.

A routing point RCOMP refines a loose route, received in an ERO, in the current domain such that all nodes within the current domain are identified. In doing so, RCOMP may notice that the ERO contains the domain exit point from the current domain. In this situation, local domain visibility allows RCOMP to determine a constrained path to that exit point; thus, the ERO can be refined and processed within the current domain using the normal procedure given in [3]. If, however, RCOMP discovers that the ERO does not include the exit point from the current domain, it uses crankback signaling with an auto-discovery mechanism. In auto-discovery, a domain exit point that can lead to the egress node is found using IGP and BGP. This mechanism is used when the egress node is not reachable using an intradomain path and the exit point from the current domain cannot be deduced from the ERO, a situation that happens when the routing points have only local domain visibility.

When the ERO does not include the exit point from the current domain, the next hop specified in the ERO will be either a loose hop DBR or an abstract node. In this case a routing point RCOMP needs to contact a TE database (TED), which is simply a database that collects, maintains, and updates the TE information (e.g., link attributes) obtained from routing advertisements. The routing point RCOMP consults a locally maintained TED and takes an appropriate action from among those listed below.

Case 1: Next hop is NOT present in TED. In this case the routing point RCOMP verifies IP reachability for the next hop (a loose hop DBR or an abstract node) using a routing table built from IGP or BGP information and static route configuration, and a *PathErr* message is generated if the next hop is not reachable. Since the next hop is not present in the TED, the next hop does not belong to the current domain. Once RCOMP establishes that the next hop is outside the current domain and is reachable, the auto-discovery mechanism is invoked to find the domain exit point (a DBR on the path to the egress node) using routing table information. The auto-discovery mechanism may provide multiple domain exit point options from which the next hop can be selected. If a path cannot be successfully found using a given domain exit point, crankback signaling extensions are used for retrying with alternate domain exit points.

Case 2: Next hop is present in TED or already found using case 1. In this case RCOMP checks local policy as well as the policy signaled by the ingress node about the LSP setup option, that is, to establish the LSP as either a stitching, nesting, or contiguous LSP. Subsequently, an intradomain part of the interdomain LSP is established based on those policies. That is, if the policies suggest that the LSP should be contiguous, RCOMP computes a path that satisfies a set of constraints and performs the ERO expansion based on the computed path. Otherwise, if RCOMP is a candidate for intra-area H-LSP (or S-LSP) setup, it selects an existing H-LSP (or S-LSP) that satisfies the TE constraints. If no such H-LSP (or S-LSP) exists, RCOMP may dynamically compute a new H-LSP (or S-LSP).

In both cases, if RCOMP is unable to compute a path through any of the exit points of its domain, it sends a *PathErr* message to an upstream routing point and includes appropriate crankback information. Otherwise, it sends a *Path* message to a downstream routing point or the egress node.

The Computation While Switching Scheme

The per-domain path computation scheme mentioned above chooses the *first* available interdomain path that fulfils the TE constraints. Thus, the resulting path could potentially be the worst possible available path. This limitation of selecting the first path in standard per-domain path computation can be overcome by using Computation While Switching (CWS), proposed in [2]. Unlike the standard per-domain path computation scheme, the CWS path computation scheme continues the quest for a better path instead of terminating the search at the first available path, by making use of a few additional crankback signaling attributes. The CWS scheme uses simple extensions to crankback signaling attributes while maintaining RSVP-TE scalability. Furthermore, it provides a mechanism to select from a set of candidate paths, each of which traverses the minimum number of domains. Thus, the CWS scheme can guarantee that the resulting path traverses the minimum number of domains. Finally, the CWS scheme exhibits path setup latency similar to that of standard per-domain path computation given in [1].

New Crankback Attributes — The key differentiating feature of CWS is the inclusion of *KT-FLAG* (keep trying) and *SKT-FLAG* (success with keep trying) attributes, which opens up the possibility of finding a "better" path even after an interdomain path has been successfully found. The *KT-FLAG* is part of the *Path* message; its value is set by the ingress node and remains constant when the *Path* message crosses the domain. When *KT-FLAG* is set, the downstream routing points, which normally do not generate a *PathErr* message if a path is successfully found, will still produce a *PathErr* message when they see the *KT-FLAG* set in the *Path* message. The *SKT-FLAG* is part of the *PathErr* message which helps the upstream routing point determine whether the downstream routing point was successful or not in finding a path. Details of additional crankback attributes introduced by CWS are given in [2].

The CWS scheme is similar to the standard per-domain path computation scheme in using the LSP setup options (i.e., stitching, nesting and contiguous) as outlined in [1]. However, the presence of *KT-FLAG* and *SKT-FLAG* allows the CWS scheme to search for an improved path even after an interdomain path is successfully found.

CWS Signaling Overhead and Latency — The CWS scheme looks for better paths, while the standard per-domain scheme given in [1] stops after the first successful path is found. The trade-off is in the signaling overhead and path setup latency, in the sense that the CWS scheme, in the worst case, could produce $O(n^2)$ RSVP *Path* and *PathErr* messages from a domain, where n is the number of domain border routers in that domain. It must, however, be pointed out that this latency is catered TO by allowing the data flow as soon as the first feasible path is found. Furthermore, the number of propagated messages may be reduced by using an optional stop-and-wait procedure that accumulates all RSVP messages directed to a neighboring domain and generates a single RSVP message; this reduces the number of RSVP messages in a domain to $O(\gamma^2)$, where γ is the number of domains directly connected to this domain.

CWS Stopping Criteria — The CWS scheme looks for better paths; if it has already found a path of length L, it only looks for paths shorter than L. That is, CWS stops the search along a potential path when the length of that path exceeds L, thus speeding up the search process. Furthermore, when the ingress node has already used all of its DBRs (exit points) for searching a best path for a given request, the search stops. A network administrator who has an idea of the maximum potential length of a valid path can also control when a search should stop.

Evaluation Methodology

We now summarize how the standard per-domain path computation scheme compares with CWS by using simulations on a real-world point of presence (POP) network. For this comparison, the primary focus is on determining the optimality of interdomain paths; thus, the intradomain topology within each POP is less significant. Furthermore, since PCE-based schemes are architecturally different from per-domain path computation schemes, a direct comparison is not attempted. For comparing CWS with the standard per-domain path computation scheme, we consider a multidomain network where each domain has on average n DBRs, m nodes, and l bidirectional links. We assume that the domains are nonoverlapping and there are no subdomains. We further assume that while computing a path, the procedure of detecting and avoiding interdomain loops is employed [5].

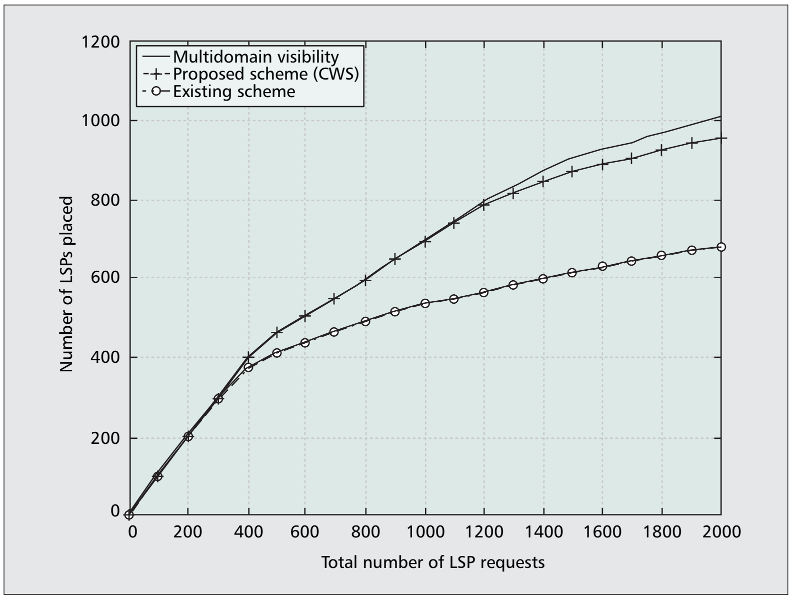
The key differentiating feature of CWS is the inclusion of KT-FLAG and SKT-FLAG attributes, which opens up the possibility of finding a "better" path even after an inter-domain path has been successfully found.

LSP requests arrive one by one in an online fashion at an ingress node such that there is no a priori information about future requests. The kth LSP request is characterized by ingress node s_k, egress node d_k, and associated bandwidth demand b_k. The ingress and egress nodes for each LSP request may or may not belong to the same domain. When they belong to different domains, the resulting LSP is an interdomain one. Furthermore, the resulting LSP between two nodes belonging to the same domain may still be an interdomain LSP if the LSP has to cross the domain boundary due to insufficient TE resources to set up an intradomain path.

In order to serve an LSP request, a bandwidth guaranteed LSP must be set up using RSVP-TE and its crankback extensions. Our main goal is to find near-optimal interdomain bandwidth guaranteed paths using a per-domain path computation scheme for the set of LSPs that cannot be confined within a domain. An additional, and important, goal is that our path computation scheme should not violate the interdomain confidentiality requirements mentioned earlier and should remain scalable. We assume that ingress has only local domain visibility, which implies that it does not have topological or resource information other than its own domain. In summary, we need to find scalable optimal or near-optimal interdomain TE paths subject to confidentiality constraints.

Simulation Experiments

We carried out our simulations for the two per-domain path computation schemes on a multidomain topology. For evaluation of interarea path computation schemes, we need two levels of network topology: a POP level backbone topology, for which we used the COST266 network consisting of 28 POPs (domains) and 37 interdomain links; and an intradomain topology, for which we used a 15-node random network in which each node pair is connected with 0.5 probability. One of the edge nodes (DBRs) of the intradomain topology was randomly chosen to establish an interdomain link with another randomly chosen DBR of similar topology. Intradomain links were chosen to have a capacity uniformly distributed between 150 and 300 units, while interdomain links were assigned a capacity of 500 units.



■ **Figure 3.** *Number of LSPs placed on COST266 POP topology.*

The traffic matrix in our evaluation process consisted of a series of LSP requests along with their required bandwidth. For each LSP request, ingress and egress nodes were randomly chosen from among all the nodes in the complete network. The bandwidth demand for each LSP request was uniformly distributed between 5 and 25 units with an infinite call holding time. LSP requests arrive one by one, and if a bandwidth guaranteed path is found, the LSP request is accepted; otherwise, it is rejected, and the network is restored to the state it was in before the arrival of that LSP. As mentioned earlier, an intradomain TE LSP may be set up if both ingress and egress are in the same domain and there are sufficient TE resources (bandwidth, in this case) to satisfy the LSP request within the domain. Otherwise, an interdomain route has to be found.

The performance of both per-domain path computation schemes we evaluated was measured in terms of the average number of domains crossed per LSP and the total number of LSPs that could be placed on a given network from a randomly generated traffic matrix. The results shown in Fig. 3 represent an average taken from 10 experiments, for each of which a new traffic matrix was generated. The results show that for the given multidomain COST266 topology of 420 nodes, the CWS scheme places about 40 percent more LSPs than the scheme in [1], while performing very close to multidomain visibility. Furthermore, we also noticed (not illustrated in the figure) that each LSP traverses 20 percent fewer domains on average when CWS is used than are traversed when the per-domain path computation scheme of [1] is used. Similar improvements were observed using the CWS for various other topologies, including the Polish POP-to-POP topology with 12 nodes and a homogeneous network topology representing the Delaunay triangulation for the 20 largest metropolitan areas in the continental United States, as reported in [2].

Conclusions

We provide a review of interdomain path computation and setup schemes for label switched networks. Such schemes are primarily categorized into PCE-based and per-domain path computation schemes. We also provide a comparison of two per-domain path computation schemes whose performance was evaluated by running a series of simulations on the COST266 interdomain topology of 28 POPs with 15 nodes per POP. One of the two per-domain path computation schemes is the newly introduced CWS scheme which, by the inclusion of a few new attributes to existing signaling procedures [1, 2], keeps finding a better path after successfully finding an initial interdomain path, thus resulting in an optimal or near-optimal interdomain path without assuming the availability of complete topology information.

The CWS scheme conforms to practical constraints of routing domains whereby the service providers do not leak routing information outside their domains for confidentiality and scalability reasons. The CWS scheme inherently guarantees that the computed interdomain paths will traverse a minimum number of routing domains by using a simple procedure to select a path among a set of candidate paths while ensuring that the domain information remains confidential. In terms of path setup latency, the CWS scheme remains comparable to other per-domain path computation schemes. The CWS scheme introduces some extra signaling whose overhead can be reduced by an optional procedure used within the CWS scheme.

We showed by simulation that the CWS interdomain path computation scheme exhibits significant improvements in terms of the total number of LSPs successfully placed and the number of routing domains traversed by an LSP. Consequently, fewer resources are utilized in the network, resulting in placement of larger traffic demands. For our simulations, we used a multidomain network of 420 nodes with random ingress-egress pairs and random bandwidth demands. The average across 10 repetitions of the experiment indicated that the standard per-domain computation scheme places fewer than 700 of the requested 2000 paths, while the CWS scheme places more than 950 paths on the same network. We also observed that per-domain computation schemes are architecturally different from PCE-based computation schemes and therefore cannot be compared directly. The choice between a per-domain or a PCE-based scheme is driven by service provider requirements, available network resources, and the network architecture.

References

[1] J.-P. Vasseur, A. Ayyangar, and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)," Internet draft, Apr. 2007, work in progress.

[2] F. Aslam *et al.*, "Inter-Domain Path Computation using Improved Crankback Signaling in Label Switched Networks," *Proc. IEEE ICC '07*, Glasgow, U.K., June 2007.

[3] D. Awduche *et al.*, "RSVP-TE: Extensions to RSVP for LSP Tunnels," RFC 3209, Dec. 2001.
[4] R. Zhang and J.-P. Vasseur, "MPLS Inter-Autonomous System (AS) Traffic Engineering (TE) Requirements," RFC 4226, Nov. 2005.
[5] A. Farrel, J.-P. Vasseur, and A. Ayyangar, "A Framework for Inter-Domain MPLS Traffic Engineering," RFC 4726, Nov. 2006.
[6] A. Farrel *et al.*, "Crankback Signaling Extensions for MPLS and GMPLS RSVP-TE," RFC 4920, July 2007.
[7] K. Kompella and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)," RFC 4206, Oct. 2005.
[8] A. Ayyangar, K. Kompella, J.-P. Vasseur, and A. Farrel, "Label Switched Path Stitching with Generalized MPLS Traffic Engineering," Internet draft, Apr. 2007, work in progress.
[9] A. Farrel *et al.*, "Encoding of Attributes for Multiprotocol Label Switching (MPLS) Label Switched Path (LSP) Establishment Using Resource ReserVation Protocol-Traffic Engineering (RSVP-TE)," RFC 4420, Feb. 2006.
[10] A. Farrel, J.-P. Vasseur, and J. Ash, "A Path Computation Element (PCE)-Based Architecture," RFC 4655, Aug. 2006.
[11] J.-L. L. Roux *et al.*, "OSPF Protocol Extensions for Path Computation Element (PCE) Discovery," Internet draft, June 2007, work in progress.

Biographies

FAISAL ASLAM completed his M.S. in computer science from the Lahore University of Management Sciences (LUMS), Pakistan, in 2004. He is currently pursuing a Ph.D. degree at Freiburg University, Germany. He is interested in network systems and sensor applications. He has also worked as a software engineer at various organizations.

ZARTASH AFZAL UZMI (zartash@lums.edu.pk) is an associate professor of computer science at LUMS. He received his B.Sc. degree from UET, Taxila, and M.S. and Ph.D. degrees from Stanford University, all in electrical engineering. He works in communications and networking with a particular focus on MPLS traffic engineering and restoration routing, connection preemption in multiclass networks, routing protocols for infrastructure and ad hoc networks, and signal processing for communications. Previously, he held positions at Bell Labs and Nokia Research Center. His current research is on aggregation mechanisms for routing protocols.

ADRIAN FARREL is managing director of Old Dog Consulting and CTO of Aria Networks, manufacturing sophisticated next-generation network modeling and optimization tools for MPLS, GMPLS, and IP networks. He is co-chair of the IETF's CCAMP, PCE, and L1VPN working groups, co-author of many MPLS-related RFCs and Internet drafts, and author of two books: *The Internet and Its Protocols: A Comparative Approach* (2004) and *GMPLS: Architecture and Applications* (2005).