1. Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 except that the right to produce derivative works is not granted.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

2. Abstract

An ISIS/OSPF routing domain may consists of multiple areas. This document postulates a set of mechanisms, and then outlines how these mechanisms could be used to establish/maintain Traffic Engineering LSPs that span multiple areas.
3. Set of mechanisms

In this section we postulate a set of mechanisms that could be used to construct LSPs that span multiple ISIS/OSPF areas. The actual application of these mechanisms to construct such LSPs is covered in the next section. We assume that the mechanisms listed below are either already available, or could be realized via extensions to the existing signaling (RSVP/CR-LDP) and/or routing (ISIS/OSPF) protocols used for MPLS Traffic Engineering.

In this document we use the OSPF term "backbone area". In the context of ISIS this term means the set of L2 routers. We also use the OSPF term "ABR". In the context of ISIS this terms means a router that is both L2 and L1.

3.1. Passing constraints

An LSR that acts as a head-end of an LSP should be able to pass the constraints associated with this LSP to some other node. The other node may, or may not be along the path taken by the LSP. Note that the existing signalling (RSVP/CR-LDP) already provides a way to pass some of the constraints, like resource affinity. [Vasseur] proposes additional extensions to RSVP to pass additional constraints. A more general way to provide this functionality is described in [Kompella].

3.2. Loose hops

An LSR should be able to specify loose ERO hops, and let some intermediate LSR(s) along the path to expand it to strict hops. Note that the ability to specify loose hops is already available.

3.3. Path computation server

An LSR that originates an LSP should be able to "ask" some other node to compute the path for the LSP. The node that computes the path may, or may not be along the path taken by the LSP. Depending on the amount of the topology and TE information available to the node, the node may compute either the whole path, or a segment of the path. In the latter case the computed path would include loose hops.

This mechanism requires the ability of the LSR that originates the LSP to pass the constraints associated with that LSP to the node that is going to compute the path for the LSP. It also requires the ability of the LSR that originates the LSP to pass the address of LSR at the tail-end of the LSP to the node that is going to compute the
path for the LSP, and for the node that is going to compute the path for the LSP the ability to pass back to the node that originates the LSP the ERO that contains the results of the path computation.

Once the path computation server computes the path, if the server is not along the path, the server is no longer responsible for maintaining the feasibility of the path (for one thing, the server may not even know whether the LSR established the path in the first place).

A path computation server usually has the topology and TE information of more than one area, including the backbone area. An example of a node that could act as as a path computation server and meet this requirement would be an OSPF Area Border Router (ABR), as an ABR has the topology and TE information of the backbone area, plus all the other areas connected to that ABR. An extreme case of a path computation server is a node that has the topology and TE information of the whole ISIS/OSPF routing domain.

3.4. Acquiring topology/TE information for multiple areas

It could be possible for a node that originates an LSP to obtain the topology and TE information of not just its own area, but other areas as well. This information may be subject to filtering (e.g., the node could obtain only the information about FAs in other areas). The node should be able to perform Constrained SPF (CSPF) based on this information.

One possible scenario is when the node, in addition to the topology and TE information of its own area, also obtains the topology and TE information of the backbone area.

An extreme case is when a node obtains the topology and TE information of the whole OSPF/ISIS domain.
4. Constructing multi-area TE LSPs

Consider the situation where the head-end of a TE LSP is in a different ISIS/OSPF area than the tail-end of a TE LSP. We’ll refer to the area that the head-end is in as the head-end area. We’ll refer to the area that the tail-end is in as the tail-end area.

Given the set of the mechanisms outlined in the previous section, the following sub-sections outline some of the possible scenarios that use these mechanisms in order to construct TE LSPs that span multiple OSPF/ISIS areas.

In the context of this document we define an optimal route as a route that would be computed in the absence of areas. In other words, an optimal route in a given network that is composed of a single OSPF/ISIS routing domain is a route that is computed if the network isn’t partitioned into areas.

It is easy to construct examples that show that partitioning a network into areas, and the resulting loss of routing information may lead to sub-optimal routing, in a sense that the computed routes would be different than optimal routes. That is true irrespective of a particular scheme to aggregate/abstract/summarize routing information. One corollary of this is that it is impossible to guarantee optimal routing in the presence of routing information aggregation/abstraction/summarization.

4.1. Scenario 1

Path computation is done on a per area basis. The head-end LSR computes strict hops within its own area. The head-end LSR then initiates LSP path setup. The setup includes the information about the constraints associated with the LSP.

The ABR in the head-end area uses the topology and TE information of the backbone area, as well as the information about the constraints associated with the LSP (the ABR receives this information as part of the LSP setup) to compute strict hops within the backbone area to the ABR in the tail-end area.

The ABR in the tail-end area uses the topology and TE information of the tail-end area, as well as the information about the constraints associated with the LSP (the ABR receives this information as part of the LSP setup) to compute strict hops within the tail-end area from itself to the tail-end LSR.

Note that since the choice of ABR in the head-end area is determined
by only the information in the head-end area, the inability to find a route across multiple areas doesn’t mean that the route doesn’t exist – it may mean that another ABR in the head-end area should be chosen, and/or it may mean than another ABR in the tail-end area should be chosen. Thus the failure to find a route may require to try another ABRs. The total number of such attempts could be as large as H*T (where H is the number of ABRs in the head-end area, and T is the number of ABRs in the tail-end area).

To exert control over which ABRs a particular LSP can traverse, the LSR that originates this LSP could be preconfigured with the addresses of these ABRs.

In this scenario the only information that has to be distributed across area boundaries are the reachability information associated with routers’ interface addresses (including loopback addresses, if an LSP is to be terminated on a router loopback address). Both OSPF and ISIS already provide mechanisms to accomplish this.

Handling link/node failures could be done as a two-phase approach, where in the first phase a failure is handled within an area where the failure occurs, and only if that fails, the head-end LSR is notified of the failure and is expected to handle it. To direct failure notifications to the ABR of the area where the failure occurs (rather than to the head-end LSR), the ABR should use the Notify Request Object.

Alternatively, link/node failures could always be handled by the head-end LSR.

4.2. Scenario 2

The head-end LSR requests an ABR in its (head-end) area to compute the path all the way from the LSR to the ABR in the tail-end area. This is possible because the ABR in the head-end area maintains the topology and TE information both for the head-end area and for the backbone area. The head-end LSR then initiates the LSP setup using the computed path. As part of the LSP setup, the ABR in the tail-end area then computes the rest of the path.

Alternatively, the head-end LSR, after requesting and obtaining the path from the ABR in the head-end area, may request the ABR in the tail-end area (which is known from the path computed by the ABR in the head-end area) to compute the path to the destination. Once this is done, the head-end LSR would have a complete path to the tail-end LSR and could use it for the LSP setup.
Note that the ABR in the head-end area that computes the path all the way to the ABR in the tail-end area may, or may not be on the path taken by the LSP.

In this scenario the only information that has to be distributed across area boundaries are the reachability information associated with routers’ interface addresses (including loopback addresses, if an LSP is to be terminated on a router loopback address). Both OSPF and ISIS already provide mechanisms to accomplish this.

In this scenario the head-end LSR may end up in a situation, where the head-end LSR, after requesting and obtaining the path that spans both the head-end area and the backbone area from the ABR in the head-end area, may find out that there is no feasible path from the ABR in the tail-end area to the destination. A way to handle this is for the head-end LSR to issue another request to the ABR in the head-end area, but indicate in the request that the ABR in the tail-end area (the one that was selected by the previous request) should be excluded from the path (this could be done by adding an additional constraint to the request).

This scenario is likely to reduce the number of LSP setup failures, relative to the scenario 1. This is because in contrast with the first scenario, there is no need to try multiple ABRs in the head-end area.

Handling link/node failures could be done as a two-phase approach, where in the first phase a failure is handled within an area where the failure occurs, and only if that fails, the head-end LSR is notified of the failure and is expected to handle it. To direct failure notifications to the ABR of the area where the failure occurs (rather than to the head-end LSR), the ABR should use the Notify Request Object. Note also, that the ABR will be informed of the failure by the routing protocol (ISIS/OSPF).

Alternatively, link/node failures could always be handled by the head-end LSR.

4.3. Scenario 3

The head-end LSR obtains the TE information from the backbone area, and uses it to compute the path all the way to the ABR in the tail-end area. The head-end LSR then initiates the LSP setup using the computed path. As part of the LSP setup, the ABR in the tail-end area then computes the rest of the path.

Alternatively, the head-end LSR, after computing the path to the ABR
in the tail-end area, may request the ABR in the tail-end area the
head-end area) to compute the path to the destination. Once this is
done, the head-end LSR would have a complete path to the tail-end LSR
and could use it for the LSR setup.

A variation on the above alternative is for the head-end LSR to
request the ABR in the tail-end area to compute paths to the
destination from all/some of the ABRs in the tail-end area. Once
this is done, the head-end LSR could use this information to compute
a path through the head-end and the backbone areas. This variation
could improve path optimality and reduce the probability of the LSP
setup failure, relative to the other alternatives described in this
section. On the other hand, it deterministically (not
probabalistically) increases the computational overhead on the ABRs,
relative to the other alternatives described in this section.

If the head-end area contains LSRs that don’t originate LSPs, then
these LSRs need not maintain the TE information from the backbone
area.

This scenario is similar to scenario 2, except for the entity that
computes path through the head-end and the backbone areas, and the
amount of information that the originating LSR has to maintain. With
respect to the amount of information, the originating LSR maintains
the same information as the ABR in the head-end area.

Handling link/node failures could be done as a two-phase approach,
where in the first phase a failure is handled within an area where
the failure occurs, and only if that fails, the head-end LSR is
notified of the failure and is expected to handle it. To direct
failure notifications to the ABR of the area where the failure occurs
(rather than to the head-end LSR), the ABR should use the Notify
Request Object.

If the head-end LSR is notified of a failure, and the failure is
neither at the ABR in the tail-end area, not in the tail-end area
itself, the head-end LSR should compute a (feasible) path to the same
ABR in the tail-end area, as the one that was used prior to the
failure. If the head-end LSR is notified of the failure, and the
failure is either at the ABR in the tail-end area, or at the tail-end
area, the head-end LSR should compute a (feasible) path to some other
ABR in the tail-end area.

Alternatively, link/node failures could always be handled by the
head-end LSR.
4.4. Scenario 4

The head-end LSR requests one of the ABRs in the head-end area to compute a path to the destination. When the ABR in the head-end area receives the request, the ABR requests one of the ABRs in the tail-end area to compute paths to the destination from all/some of the ABRs in the tail-end area. Once the ABR in the head-end area obtains this information from the ABR in the tail-end area, the ABR in the head-end area could compute the path through the head-end and the backbone areas, concatenates the results of this computation with the appropriate path through the tail-end area, and then return the result to the head-end LSR.

Note that the ABR in the head-end area that computes the path all the way to the ABR in the tail-end area may, or may not be on the path taken by the LSP.

Likewise, the ABR in the tail-end area that computes the path(s) in the tail-end area may, or may not be on the path taken by the LSP.

In this scenario, just like in the Scenario 2, the only information that has to be distributed across area boundaries are the reachability information associated with routers' interface addresses (including loopback addresses, if an LSP is to be terminated on a router loopback address). Both OSPF and ISIS already provide mechanisms to accomplish this.

While this scenario is likely to reduce the probability of the LSP setup failure relative to the Scenario 2, it does this at the expense of deterministically (and not probabilistically) higher computational overhead placed on ABRs.

Note that the approach outlined in this scenario results in computing optimal routes.

The approach described in this section could reduce the probability of the LSP setup failure, relative to the approach outlined in Scenario 2. On the other hand, it deterministically (not probabilistically) increases the computational overhead on the ABRs, relative to the approach outlined in Scenario 2.

The approach described in this section could also facilitate computation of node/link disjoint paths.
4.5. Scenario 5

The head-end LSR requests an entity that has the entire network (all areas) topology to compute the whole path.

Except for the entity that has the entire network topology, in this scenario the only information that has to be distributed across area boundaries are the reachability information associated with routers’ interface addresses (including loopback addresses, if an LSP is to be terminated on a router loopback address). Both OSPF and ISIS already provide mechanisms to accomplish this.

Note that the approach outlined in this scenario results in computing optimal routes.

Handling link/node failures could be done as a two-phase approach, where in the first phase a failure is handled within an area where the failure occurs, and only if that fails, the head-end LSR is notified of the failure and is expected to handle it. To direct failure notifications to the ABR of the area where the failure occurs (rather than to the head-end LSR), the ABR should use the Notify Request Object.

If the head-end LSR is notified of a failure, the head-end LSR should request the entity that computed the whole path to compute another path. When issuing such a request, the LSR should specify that the path should exclude the element that caused the failure.

The approach described in this section could also facilitate computation of node/link disjoint paths.

5. Pre-engineered backbone area

In certain cases it may be desirable to "pre-engineer" the backbone area by constructing a set of TE LSPs that would be used as FAs by the traffic that has to traverse the backbone area. The scenarios outline above do not preclude this as an option.

With a pre-engineered backbone area, a variation of Scenario 3 would be to restrict the backbone information leaked to the non-backbone areas to only the information associated with the FAs in the backbone area.

Note that an LSP that starts/ends in a non-backbone area, but traverses the backbone area may use more than one FA. That means, for example, that even if there is no FA between any of the ABRs in the head-end area and any of the ABRs in the tail-end area, an LSP could
still be established, as long as there are some other FAs that combined together provide a path from the head-end to the tail-end area.

Stability of the LSPs that are used as FAs in the backbone area is unaffected by the establishment/tear down of the LSPs that start/end in other areas, but traverse the backbone area.

Pre-engineering the backbone area enables aggregation of the Label Forwarding State in the backbone area, as a TE LSP that is used as an FA could carry/nest multiple LSPs. One possible application of this feature would be in conjunction with GMPLS controlling OXCs, where OXCs would be placed in the backbone area.

6. TE information summary

One possible application of summarization of TE information would be to prune off unfeasible ABRs, as follows. An ABR in the tail-end area would compute two paths: (a) a min hop path to the destination ignoring all other constraints (such as b/w or resource affinity), and (b) a max unreserved b/w path to the destination, again ignoring all other constraints (such as resource affinity or unreserved b/w).

The ABR will then advertise the hop count from the first path and the max unreserved b/w from the second path into the backbone area. The ABR in the head-end area will do exactly the same, and then advertise this information into the head-end area. An LSR in the head-end area that needs to originate an LSP could use this information to prune off unfeasible ABRs. For example, if the LSP requires X amount of bandwidth, the LSR should rule out any of the ABRs in its area that advertise less than X amount of unreserved b/w to the LSP’s destination.

While pruning off unfeasible ABRs, as outlined in the previous paragraph, could reduce the number of LSP setup failures, it is not without drawbacks. For one thing, it increases the amount of control (routing) information that an LSR has to maintain, as well as the volume of the control information that an LSR receives from other LSRs, and has to process. In addition, it adds computational overhead to ABRs, as they need to perform additional computation. Moreover, establishment/tear down of inter-area LSPs increases the frequency with which this computation would need to be performed. Finally, it is important to realize that this approach doesn’t eliminate LSP setup failures, as for one thing, the information computed and advertised by ABRs may be "out-of-date". Yet another reason for LSP setup failures is that the unreserved b/w information computed by ABRs doesn’t take into account any other constraints, while a particular LSP may be subject not just to unreserved b/w, but to
other constraints (e.g., resource affinity) as well. So, even if from the unreserved b/w constraint point of view a particular ABR is feasible for a particular LSP, when taking into account all the constraints associated with that LSP, the ABR may no longer be feasible.

And to reiterate the point made earlier, since route computation with this approach uses summarized/aggregated/abstracted information, this approach doesn’t guarantee optimal routing.

7. Protection in the multi-area environment

An LSP that spans multiple area could be protected in a variety of ways. One alternative is to establish a backup (secondary) LSP, and make sure that this LSP is node/link disjoined from the (primary) LSP that it protects.

Another alternative is to provide protection on a per-area basis, where LSP’s segment that is within a given area is protected by another LSP (this other LSP protects just that segment). Note that by using the label stacking construct a single LSP could be used for protecting segments of multiple other LSPs.

Finally, within each area one could employ the same protection mechanisms as are currently used for LSPs that are confined to a single area.

8. Reoptimization

Reoptimization is the set of mechanisms by which an existing (already established) LSP is rerouted over a more optimal path. In general reoptimization could be either periodic (timer-driven), or triggered (event-driven). An example of an event that could trigger reoptimization is coming up of a link that was previously down.

Periodic (time-driven) reoptimization is performed by the head-end LSR.

Handling event-driven reoptimization could be done on a per area basis. That is, if the event that triggers reoptimization happens in the head-end area, it is the responsibility of the head-end LSR. If the event happens in the backbone area, then it is the responsibility of the ABR in the head-end area. Finally, if the event happens in the tail-end area, then it is the responsibility of the ABR in the tail-end area.
Just like with a single area reoptimization, the multi-area reoptimization should be done without traffic disruption (by using the approach known as "make before break").

9. Security Considerations

Security issues are not discussed in this document.

10. Acknowledgements

We would like to thank Noel Chiappa, Tony Li, Robert Raszuk, and Alex Zinin for helping us with the ideas presented in this document.

11. References

[Kompella] Kompella, K., "Carrying Constraints in RSVP"


12. Author Information