INTERNET DRAFT
Category: Informational
Title: draft-thorup-ospf-harmful-00.txt
Date: April 2003

Mikkel Thorup
AT&T Labs-Research

OSPF Areas Considered Harmful

Status of this Memo

Abstract

We point out that forcing an area decomposition onto an IP backbone can easily have a detrimental effect on the routing and lead to a substantial increase of information to be flooded in connection with link-failures.

1.0 Introduction

Currently it is recommended that a larger IP backbone running OSPF should be divided into areas. We point out here that with the current definition of areas, this can easily have a detrimental effect on routing, increase the amount of information in the network, and cause many more changes to be flooded in connection with link-failures. More precisely, our concerns are (1) the special area routing and (2) the fact that area border routers communicate distances into an area rather than just exposing the

area topology. These choices appear to be made with the intent of reducing the amount of information flooded and speeding up route computation, but we will argue that they are more likely to have quite the opposite effect.

Based on the potential problems with areas, we contend that it is counter productive that network operators by default try to get OSPF to scale using an arsenal of different types of areas, i.e., areas, stub areas, and not-so-stubby areas, all adding to the complexity of configuring OSPF. If there really are scaling problems with plain OSPF without areas, the router vendors could work on more efficient implementations of plain OSPF, e.g., using incremental shortest paths to deal with link-failures.

We note that the similar IS-IS protocol is commonly used single-level, suggesting that plain OSPF is feasible as well.

2.0 Plain OSPF routing

In Open Shortest Path First (OSPF) routing [Moy98], each link has a weight, and packets follow shortest paths to their destinations. The weights may either have default values inversely proportional to their capacity or be set more carefully by the network operator. Many works [RFT02,FT00,RR01,LW93,BGW98] have demonstrated that this kind of plain shortest path routing works surprisingly well for distributing the traffic in a network.

For OSPF routers to maintain their forwarding tables, each router needs to know the whole network topology, including link weights. The routers can then locally compute shortest paths via each outgoing link to all destinations, and thereby decide on next hops for the forwarding table.

In order to provide the topology information for all routers, the links and their weights are flooded in the network, both on a regular basis and in connection with changes.

In this draft, we do not qualify the difference between routers, networks, and hosts as they are all treated the same from the perspective of routing. Abusing terminology, we refer to all of these network nodes as routers.

3.0 Division into OSPF areas

For large networks, concerns about scalability have lead to the introduction of areas that, among other things, aim at reducing the amount of information and computation. Below, we describe these areas on a high level, pointing out that they are likely to have quite the opposite effect, increasing rather than decreasing the amount of information and computation.

In an OSPF network, we can have areas 0,1,.... The area 0 is called the backbone area and has a special role. Routers may belong to one or more areas. Routers belonging to more than one area are called area border routers, whereas routers belonging to a single non-backbone area are called internal routers. All area border routers should belong to the backbone area. For the backbone area one may define virtual links over other areas, implemented as paths over these areas. Each area should be contiguous (the backbone area has to be contiguous, and if one of the other areas is not contiguous, each of its connected components is viewed as a separate area).

4.0 Area routing

When we have areas, we do no longer just follow shortest paths. Instead we apply the following rules:

1. A router in the same area as the destination only considers paths within the area.

2. A backbone router that is not in the destination area considers paths with a first segment in the backbone and a second segment in the destination area (if the destination is in the backbone, the last segment is empty).

3. An internal router not in the destination area considers paths with a first segment to the backbone, a second segment in the backbone, and a last segment in the destination area.

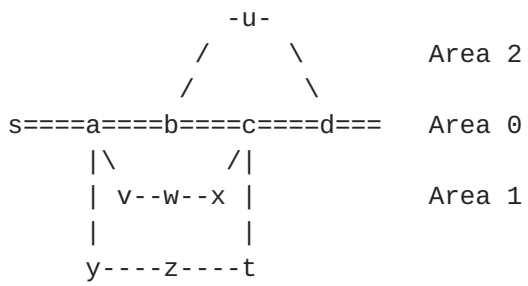

Figure 1: Area routing

4.1 Funny area routing

Figure 1 illustrates some of the peculiarities of area routing. We have backbone area 0 with high-speed high-capacity links, and some areas on either side with much smaller links. Each of these non-backbone areas has two border routers on the backbone.

Following the default of setting weights inversely proportional to

their capacity, plain shortest path routing will follow the backbone as much as possible, and this is what we want. For example, to get from source s to destination t, with plain OSPF, we will follow the route

(1) s-a-b-c-t

However, consider the area routing from s to t. The source s is not in the destination area, so we start with rule 3, and from there, we do consider the above shortest path. However, router a is a border router for Area 1 which contains our destination t. Hence, when we get to router a, we switch to rule 1 and divert on a local detour over y to t. Thus, with area routing, we get the route

(2) s-a-y-z-t.

This route is considered worse than the one in (1) because it uses local links of smaller capacity.

To make matters even worse, suppose the local link between a and y fails. This would not affect our plain shortest path route in (1), but with area routing, we would now go from a to v, hence using the route

(3) s-a-v-w-x-c-t.

In order to avoid funny area routing like above, each area A should be convex in the sense for any source s in A and destination t in A, the area A should contain all shortest paths from s to t in the full network. In the example of Figure 1, this means that each of the non-backbone areas should be extended with the segment of the backbone between its current entry points. This implies that we need a-b-c in Area 1 and b-c-d in Area 2. In particular, we end up with a logical copy of b-c in each of Area 0, 1, and 2. Even if this sharing is possible, it all adds to the complexity of the configuration. Also, getting b and c as new border routers for Area 1 and 2, respectively, adds to the information flooded in the network, as discussed in the next section.

5.0 Distances into areas

One of the main ideas behind areas is that the internal topology is not revealed to the rest of the network. Each router in the area computes shortest paths into the area, producing a distance table. The area border routers communicate their distance tables to the rest of the network, which then doesn't need to know the internal topology of the area.

5.1 Areas are likely to increase the information

With distance tables, for each internal router r in an area A, the rest of the networks sees a distance to r from each area border router of A. Thus, with distance tables, we get more information on r if the number of area border routers for A is bigger than the degree (number of incident links) of r. Often the average router has limited degree, say 4-6, and then it is limited how much we can win switching from topology to distances (areas with a single border router are kind of special, and will be discussed further in Section 9.1). On the other hand, we may lose a lot. Take, for example, a grid topology as illustrated in Figure 2.

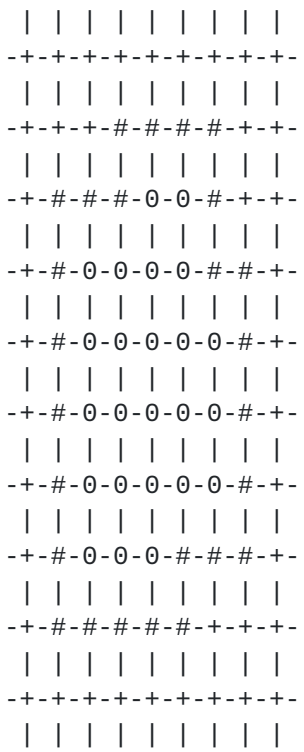

Figure 2: Area strictly inside large grid with border routers # and internal routers 0.

In a standard grid, any area with 50 routers strictly inside has at least 26 area border routers. On the other hand, in the grid, each router only has degree 4, so we are multiplying the amount of information by more than a factor 5. More generally, if an area strictly inside a grid has n routers, it has at least 4*sqrt(n)-4 area border routers, the best case being if n is a perfect square. Even then, the amount of information grows with nearly a factor sqrt(n).

For more random-like networks, the situation is even worse due to

their expander properties [Bol85]. For example, if a random graph has degree 3 or more, we expect that all possible areas have most routers on the boundary (this excludes giant areas subsuming most of the routers).

In other words, we should not expect a topology to admit any useful decomposition into areas unless the topology is designed to admit areas with few, say 1-2, area border routers. We note that such a design fits a two-level topology with areas being PoPs connected to a more highly connected backbone area. In that case, we should not expect to be able to further subdivide the backbone area. Also, we might later want to add another connection from a PoP, e.g., to make it more robust against failures, or just because a new connection is the best solution for a congestion problem. In that case, with areas, we get a new border router, hence a new distance to each router in the area. Without areas, we are just adding a link. Another problematic case for areas is if an ISP buys up other networks, merging them with its own. Then it is very likely that the higher connectivity will prevent a useful area decomposition.

In short, insisting on areas limits the design of networks, and in particular, it limits how well-connected the networks can be. Conversely, since most networks are sparse with few links per router, even if we do succeed in identifying areas with few border routers, it is very limited what we can win with the areas communicating distances instead of internal links.

6.0 The shortest path computation

It may seem that having area border routers offering distances rather than internal links should be an advantage for the single source shortest path computation that each router has to perform to fill its forwarding table. However, a simple Dijkstra shortest path algorithm with a good heap implementation spends so little time per link that we cannot gain much by processing distances instead of links. In fact, the speed of the shortest path computation should be a non-issue: for a large network with a 1000 routers and 5000 links, a standard single source shortest path computation takes in the order of a millisecond on today's computers whereas the convergence time for OSPF in connection with link-failures is in the order of seconds [AJY00].

Interestingly, in [AJY00], the shortest path computation is reported to be in the order of hundred times slower on some high-end routers, and curve fitting even indicated a quadratic time shortest path computation. This illustrates the kind of problems one gets when making OSPF more complicated with a hack like areas. Without areas, vendors could just have used one of the many near-liner time Dijkstra implementations that have circulated

for more than 30 years. However, now they had to make their own specialized solution, allowing them to come up with an unscalable quadratic implementation. When vendors can make such mistakes on expensive routers, it is troubling to think what kind of mistakes network operators might do with areas in the daily running of a network.

7.0 Link-failures starting avalanches of distance changes

One of the more scary features of distance tables is that if a link fails, then all distances using this link may get affected. For example, in Figure 1, if we get any link-failure on the path a-v-w-x-c, then each router in the path gets a new distance from one of the area border routers a and c. One could imagine much worse situations with a bottle-neck link that affected distances from many border routers to many internal destinations. Thus, a simple link-failure may lead to an avalanche of distance changes flooding the network.

One may think of it as an advantage that we do not communicate a link-failure outside an area A if it does not affect any distances into the area. However, without areas, such a link-failure is easily dealt with if we employ an incremental shortest path algorithm, such as the one in [RR96], from each outgoing link. If no distances into A are affected (without areas, we just think of A as a region of the network), then for any router outside A, no distance via any outgoing link is changed. Such an innocent link-failures is detected in a few instructions, that is, in nanoseconds, and it does not entail any changes to the routing tables.

Thus, areas may turn simple link-failures into big problems (avalanche of distance changes), and the potential savings (not reporting an innocent link-failure) are miniscule.

As reported in [AJY00,FT00], the use of incremental shortest paths algorithms [RR96] gives a large speed-up (factor 10-100) for local changes such as link-failures --- also if these do affect some distances. This is a clear advantage for plain OSPF. With areas, we can also make limited use of incremental shortest paths, but when an area creates an avalanche of distance changes, outside routers will still have to check for each distance change how it affects the choice of outgoing ports. This is hence a good example showing how the simplicity of plain OSPF allows more scalable implementation.

8.0 Summarization and infinite loops

One of the big potential wins of areas is that one can identify

interior destinations in the area with a prefix mask, an approach called "summarization". In the distance tables of the area border router, the distance to the mask will be the maximal distance to a destination represented by the mask. Obviously such masks can lead to very substantial savings depending on how many routers they capture.

We note that summarization requires rule 1 from Section 4.0, i.e., that we cannot leave the destination area. The point is that the distance to a destination may be perceived as shorter inside an area than outside an area, and any such inconsistency can lead to infinite loops if packets are allowed to move in and out of areas.

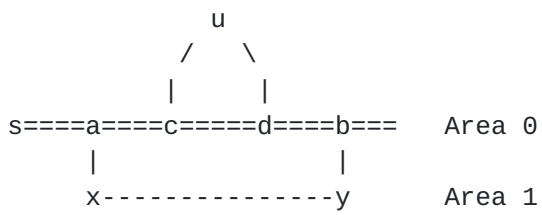

Figure 3: Summarization example.

It may seem pretty safe to use summarization on routers close to each other. However, such summarization can lead to unintended infinite loops in connection with link-failures. Consider the example of Area 1 in Figure 3. Without summarization, no single link-failure can disconnect x from y in the overall network. However, suppose we summarize the internal routers x and y in a common prefix mask m. Assuming unit weights, both area border routers a and b will export a distance of 2 to m. Now, if the link from x to y fails, it disconnects Area 1 into a part X containing a and x and a part Y containing b and y. From a, the longest distance to an address matching m in X is the distance of 1 to x. Similarly, the the distance from b to m in Y is 1. Thus, both a and b will export a distance of 1 to m. Now, consider a packet p at c destined for y. Matching y with the mask m, router c will think the distance to y is 2 via a, and forward p to a. However, a is in X and knows that y is not there, so a will send p back to c, thus creating an infinite loop. For contrast, if link x-y fails without summarization, a would only export a distance of 1 to x, and b would only export a distance of 1 to y, and then p would just follow the correct shortest path to x via d and b.

It is a very appealing quality of plain shortest path routing that no matter how many links fail, if the remaining network has a path from one point to another, then such a path will be selected by the protocol. The infinite loop example above shows that this quality is lost with summarization.

9.0 Stub areas

The basic idea in stub areas is a kind of inverse summarization, where we summarize everything outside the ISP with the empty mask, matching everything that is not matched inside the ISP. The advantage to this is that the stub area does not need to know about any routes leaving the ISP. This allows for some less powerful routers inside a stub area. The fact that the stub area does not care where we are going if it is outside the ISP can lead to more funny routing, so as with everything else related to areas, they have to be used with caution. Some not-so-stubby areas have been introduced that care about some but not all of the outside world. This can be used to give less funny routing, but it also adds to the complexity of configuring OSPF.

9.1 Leaf areas

Given all the above warnings against area routing, we note that there is one safe kind of areas; namely areas with a single border router. For such a "leaf area", we can never have any problems with the area routing. Also, for these, all we need to know from the outside is what IP addresses are inside, and vice versa. We note that potential leaf areas can be identified in linear time. More precisely, we are looking for single routers whose removal disconnect the network. These are called articulation points in graph theory and all articulation points can be identified in linear time (c.f. [CLRS01,HT73]). Having identified the articulation points, automatically, the OSPF implementation could exploit whatever advantages there are to leaf areas without the need for configuration by the network operator.

10.0 Which area to chose

If convinced that plain OSPF routings is best, we still have some decisions to make concerning which type of area to use. The most natural choice would be to put all routers in Area 0. However, a popular feature of stub areas is that routers inside a stub area cannot export BGP routes into OSPF. To exploit the above stub area feature with plain OSPF, we can put all routers in one big stub area. If we want some of the routers to behave as backbone routers, we can also put these in Area 0. The backbone routers can be connected with virtual links, but these will never be considered for routing since all routing happens within the same stub area (c.f. rule 1 in Section 4.0). This way we can freely partition routers into stub routers and backbone routers without having to deal with the pitfalls of area routing.

We note that the above is only a hack to benefit from features of different types of areas without having to deal with area

routing. A cleaner solution is to use routers where the desired features can be configured independent of area types, e.g., with a maximum on the number of BGP routes that can be exported into OSPF.

11.0 Concluding remarks

We have argued that OSPF areas are dangerous and often of very limited use. It is therefore important not to count on them for solving scaling problems in OSPF. Rather one should focus on the scaling of plain OSPF without areas, exploiting the simplicity in faster and cleaner solutions.

12.0 Acknowledgments

I would like to thank Jay Borkenhagen, Dave Katz, Carsten Lund, Jennifer Rexford, and Aman Shaikh for many good discussions on the theory and practice of OSPF areas.

13.0 References

[AJY00] C. Alaettinogly, V. Jacobson, and H. Yu.
Towards milli-second IGP convergence.
Expired internet draft. Now available as
http://www.packetdesign.com/pubs_003.html, 2000.

[BGW98] A. Bley, M. Grotchel, and R. Wessaly.
Design of broadband virtual private networks:
Model and heuristics for the B-WiN.
In Proc. DIMACS Workshop on Robust Communication
Networks and Survivability, AMS-DIMACS Series 53,
pages 1--16, 1998.

[Bol85] B. Bollobas. Random Graphs. Academic Press, 1985.

[CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein.
Introduction to Algorithms. MIT Press, McGraw-Hill,
2nd edition, 2001.

[RFT02] B. Fortz, J. Rexford, and M. Thorup.
Traffic engineering with traditional IP routing protocols.
IEEE Communications Magazine, 40(10):118--124, October 2002.

[FT00] B. Fortz and M. Thorup.
Internet traffic engineering by optimizing OSPF weights.
In Proc. 19th IEEE Conf. on Computer Communications (INFOCOM),
pages 519--528, 2000.

[HT73] M.L. Fredman and R.E. Tarjan.

Algorithm 447: efficient algorithms for graph manipulation.
Communications of the ACM, 16(6):372 -- 378, 1973.

[LW93] F.Y.S. Lin and J.L. Wang.
Minimax open shortest path first routing algorithms in networks supporing the SMDS services.
In Proc. IEEE International Conference on Communications (ICC), volume 2, pages 666--670, 1993.

[Moy98] J. T. Moy.
OSPF version 2.
Network Working Group, Request for Comments, http://www.ietf.org/rfc/rfc2328.txt, April 1998.

[RR01] K.G. Ramakrishnan and M. A. Rodrigues.
Optimal routing in shortest-path data networks.
Bell Labs Technical Journal, 6(1), 2001.

[RR96] G. Ramalingam and T. Reps.
An incremental algorithm for a generalization of the shortest-path problem. Jounal of Algorithms, 21(2):267--305, 1996.

14.0 Author's Address

Questions about this memo can be directed to:

Mikkel Thorup
AT&T Labs-Research
Florham Park, NJ 07932
USA

Phone: +1 (973) 360 8904
Fax: +1 (973) 360 8178
E-mail: mthorup@research.att.com