

# Route Flapping Effects on OSPF

Yasuhiro Ohara  
KEIO University  
Graduate School of Media and Governance  
5322 Endo, Fujisawa Kanagawa, 252-0804 Japan  
yasu@sfc.wide.ad.jp

Manav Bhatia  
Samsung Electronics Co. Ltd.  
Samung India Software Operations  
No. 67, Infantry Road, Bangalore India  
manav@samsung.com

Osamu Nakamura  
KEIO University  
Faculty of Environmental Information  
5322 Endo, Fujisawa Kanagawa, 252-0804 Japan  
osamu@wide.ad.jp

Jun Murai  
KEIO University  
Faculty of Environmental Information  
5322 Endo, Fujisawa Kanagawa, 252-0804 Japan  
jun@wide.ad.jp

## Abstract

*Route flap is an undesirable phenomenon in the Internet and needs to be eliminated for more stable and robust networks. In this paper we present our observations of such persistent route flaps on OSPF in some detail. We show how flaps adversely affect OSPF routing and communication environment in general through our experiments. The current OSPF's implicit timer damping function is found inefficient as each flap may lead to a few seconds of lost connectivity. We implemented a scheme for damping the route flaps and show as to how this methodology can solve such problems in OSPF. Our implementation shows considerable improvement over the ones where none of such techniques were applied.*

## 1. Introduction

Internet's communication data paths are controlled by routing protocols. They are responsible for finding alternative paths/routes in the face of network failure, such as hardware failures, excessive load conditions, incorrect implementations, network upgrades/routine maintenance, etc. In such cases, communication across the network can be partially, or even totally lost until the routing protocols find

the next best path for all the routes using the network that has failed.

When a network event causes routes to either go down or become available, routers distribute routing update messages that permeate networks, stimulating re-computation of optimal routes and eventually causing all routers to agree on these routes. The process from network changes to recovery of communication paths is called "convergence" wherein all the routers agree on their view of optimal paths.

During convergence, there can be routing loops in the network and a subset of destinations will be reached via sub optimal paths or will not be accessible at all. Routing loops themselves may delay the overall convergence process as there will be some routing control packets lost. Thus it is desired that the routing protocols in the routing domain take minimum time to converge for this will lead to increased network reliability as the periods for which routes will not be available/non-optimal will be minimum.

The central symptom of route instability is the disappearance of a route that previously existed in the routing table. Such routes may disappear and reappear intermittently, a condition referred to as "flapping". Thus the number of route flaps characterize the intensity of the perturbation in the network.

In this paper we study the effect of such persistent "route flaps" on OSPF[3], a link state routing protocol. We ob-

served how it disrupts the routing and adversely affects the communication environment. Further, we implemented a scheme for damping the route flaps and show as to how this methodology can solve such problems in OSPF.

## 2. Route Flapping and its effect

We explore the effect of persistent route flapping on OSPF and a UDP[5] communication channel through our experiment in this section.

### 2.1. Network Configuration

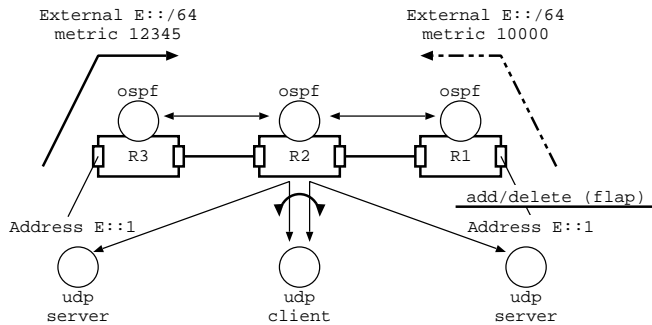


Figure 1. Network Configuration

The network setup used in our experiment is shown in the figure 1. There are three routers, each of which runs OSPFv3[2] using Zebra ospf6d[7][4]. Both R1 and R3 advertise the prefix E::/64 as AS-External route with metrics 12345 and 10000 respectively. Additionally both R1 and R3 are configured with address E::1 on their interfaces and thus each can receive packets destined to E::1.

R1 and R3 are configured as UDP servers while R2 runs the UDP client sending a stream of test UDP packets. These packets will be received by either R1 or R3, depending upon whichever is chosen as the best path at that particular instant. Under the steady state conditions R1 is the preferred destination as it is advertising the prefix with a smaller metric.

Upon receiving the test UDP packet, the UDP server writes down the value of HopLimit (TTL), fills its hostname as means for identification and sends it back to the source (UDP client). The UDP client, by comparing the time when the packet was sent and the time when it is received, computes the total round trip time (RTT) for each packet. By examining the hostname filled in the packet it can know which router received that particular packet and can thus estimate when the traffic starts using the secondary path. By looking at the sequence numbers it can determine the number of packets which get dropped and by examining

the round trip hop count, it can know when all, the packets got stuck in a routing loop.

As described above under the steady state conditions the primary path for all traffic destined to E::1 is via R1. This is our primary path. When this path is not available the traffic shifts to the secondary path, which is through R3.

Flapping is artificially introduced in the setup by adding and deleting the address E::1 to/from R1's interface using the ifconfig command in NetBSD[6]. Zebra ospf6d is dynamically notified of this event and it in turn advertises this information to both R2 and R3 by originating and purging the corresponding AS-External-LSA.

### 2.2. OSPF behavior

There are 2 built-in fixed timers in OSPF which implicitly damp the origination of LSAs against flapping entities. The first is MinLSInterval that states that an LSA cannot be originated within 5 seconds if a previous copy was originated within this time period. The other is MinLSArrival timer that does not accept newer LSAs if a previous copy was received within 1 second back. These two timers, can however, adversely affect in certain circumstances as we shall see.

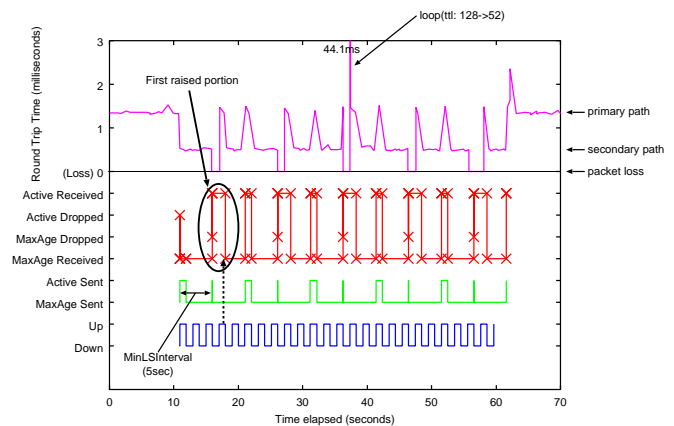


Figure 2. RTT, OSPF send/receive event during flapping of 2 seconds cycle (1 second for Up, 1 second for Down)

The figure 2 illustrates the relationship between the periodic Up/Down event, OSPF send/receive event for LSAs, and reachability of UDP communications. In the figure, the lowest line in the chart indicates the event causing the network topology change, which is in this case, addition and deletion of address E::1 at R1. For the sake of brevity we shall call adding the address E::1 to R1's interface as the Up event and deleting it as the Down event. Each

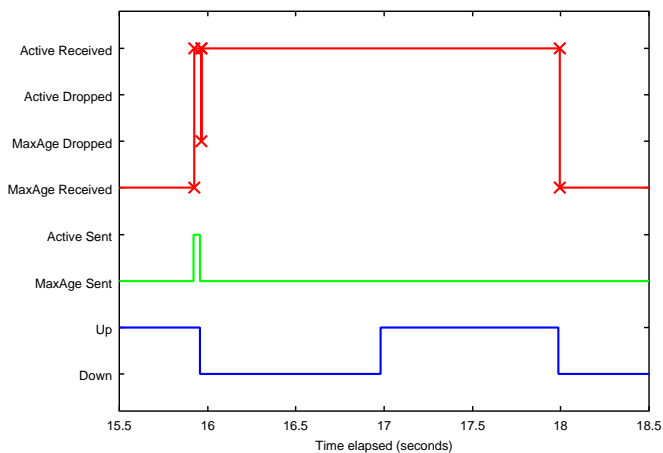
*Up/Down* event is generated periodically after every 1 second as shown in the figure.

The second line from bottom shows the origination of appropriate LSA for prefix *E::/64* at *R1*. Each LSA can only be originated after every 5 seconds as the LSA origination interval is limited by *MinLSInterval* seconds in the OSPF specification. Thus at each *Up* event, *ospf6d* tries to originate the corresponding AS-External LSA with age 0 unless affected by the *MinLSInterval* (described as “*Active Sent*” in Figure 2). At each *Down* event, *ospf6d* tries to flush this LSA from the routing domain by flooding a copy of this LSA with its LS Age set to *MaxAge* (3600) (described as “*MaxAge Sent*” in the figure). We often observed that the LSA origination was immediately followed by attempts at flushing it. This is caused by the timing of the *Down* event.

The third from the bottom illustrates the corresponding AS-External-LSA receive events at *R2*. There are four receive events: *MaxAge Received*, *MaxAge Dropped*, *Active Dropped* and *Active Received*. As the names suggest, the events *Active Dropped* and *MaxAge Dropped* indicate the event when *ospf6d* rejects the received LSA because of the *MinLSArrival* restraint while *MaxAge Received* and *Active Received* indicates the event when the LSAs are accepted.

The top line in the chart indicates the total round trip time (RTT) between the *UDP client* and the *UDP server*. The packets which are lost during the re-route period are shown with RTT value of 0.

In the same figure, the *first raised portion* of OSPF receive events, shows the adverse effects of the *MinLSInterval* and *MinLSArrival* timers. The scale changed figure 3 illustrates the part in detail. It is explained as follows.



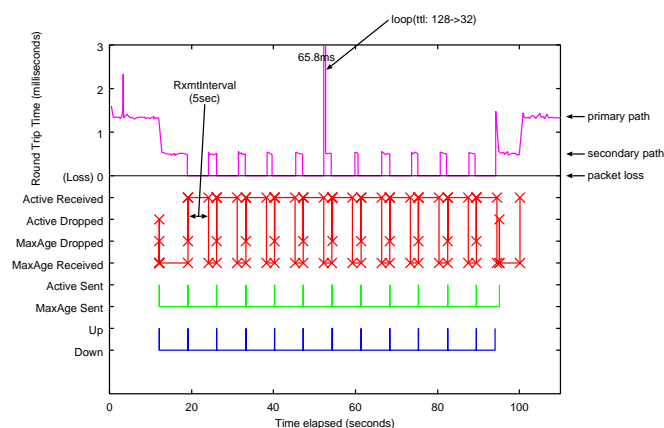
**Figure 3. The first raised portion of the figure 2**

When receiving the LSA that corresponds to *Active Sent* (which was delayed by *R1* due to *MinLSInterval*), the event

*Active Received* occurs. Then, as mentioned before, the event *MaxAge Sent* gets triggered due to the real *Down* event immediately after the *Active Sent* event. Since OSPF specifies that the LSA be dropped if a previous copy is received less than *MinLSArrival* (1 second) ago, this *MaxAge* LSA flood is rejected by *R2*.

This results in routing disparity where different routers have different views of the optimal paths and the packets get dropped in the test UDP communication network, as *R2* still believes that *R1* is the best path for *E::/64* whereas in reality it is actually down.

Zebra *ospf6d* do not apply *MinLSInterval* when flooding the *MaxAge* LSAs, and these LSAs are flooded as soon as the *Down* event occurs. The problem is that the other routers will still apply the *MinLSArrival* timer. Since the other routers had just received the LSA they ignore this *MaxAge* LSA. The faulty routing thus persists in the routing domain till either the next real *Down* event or the expiry of the Retransmission timer - whichever occurs first. In our case the event *Down* occurs first and the *MaxAge* LSA is originated again by the *R1*. This time the other routers accept this LSA and the routing gets converged and packets start flowing through the secondary path. Even if implementations apply *MinLSInterval* to control the flooding of *MaxAge* LSA the problem is only going to get worse for the period during which false routing exists will last longer.



**Figure 4. RTT, OSPF send/receive event during flapping of 7 seconds cycle (*Up* immediately followed by *Down* and remains *Down* for 7 seconds)**

The figure 4 shows the worst case scenario for this environment by executing *Up* event immediately followed by the *Down* event. with a period of 7 seconds. Thus the period of *Up* event is a little (less than 1 second) and that of the *Down* event is 7 seconds.

This result shows that all the *MaxAge Sent* triggered by *Down* event were rejected by *R2*, so the incorrect routing information lasts until *R1* retransmits its un-acknowledged LSA to *R2* after the expiry of *RxmtInterval*. In OSPF, *RxmtInterval* is a configurable interface parameter, which is set up as 5 seconds for this experiment. Here the RTT for the UDP communication channel at the top of the figure shows packet loss for 5 seconds, which then recovers to use the secondary path until the beginning of the next *Up/Down* cycle, for 2 seconds.

### 2.3. Summary of results

In this section, we summarize what we observed from our experiment.

First, the current timers in OSPF (*MinLSInterval*, *MinLSArrival*) which we think are enough to damp short flaps are inadequate and fail miserably when the period of flap becomes greater than 5 seconds. Even for flaps with periods less than 5 seconds there can be transient blackholes depending upon how the *MinLSInterval* and purging of the LSA (*Down* event) are synchronized.

During this period as OSPF advertises and calculates these false routes, there are little chances for communication using the alternative stable route even if there exists one. OSPF when used in conjunction with Traffic Engineering further aggravates the situation because each time the network topology changes, traffic engineered paths may need to be rerouted. Even if alternate paths have been recomputed, there are overheads involved in tearing down the old route and setting up the new one.

It is shown that *MinLSInterval* which delays the origination of LSA up to 5 seconds and *MinLSArrival* which delays receiving and processing LSA up to *RxmtInterval* seconds are both pathological in some particular scenarios.

By varying the period and cycles of flaps we collected some very interesting statistics. We kept the total period of flap as 2 seconds - 1 second of *Up* event followed by 1 second of the *Down* event. We observed that the UDP communication followed the following cycle. In a period of 10 seconds there was 1 second of packet loss, 1 second using primary path, 3 seconds of secondary path, 1 second of primary path and 4 seconds of secondary path (All these measurements are approximate). Thus in a cycle of 10 seconds the primary or the better path was only used for 2 seconds.

In another flap setup we kept the *Up* event for some small time ( $< 1$  second) followed by 7 seconds of the *Down* event. We observed that the UDP communication followed a period of 7 seconds in which for 5 seconds all the packets were lost and for some mere 2 seconds the packets used the secondary path.

Given the way the IGP's are used nowadays this behavior of OSPF against flaps is clearly inadequate and warrants

closer inspection.

## 3. OSPF flap damping

To solve these problems we implemented a solution very similar to one as explained in BGP Route Flap Damping[8]. We categorized paths/routes as either "well behaved" or "ill behaved". A well-behaved route shows a high degree of stability during the extended period of time. On the other hand, an ill-behaved route experiences a high level of instability in a short period of time.

### 3.1. Flap damping algorithm

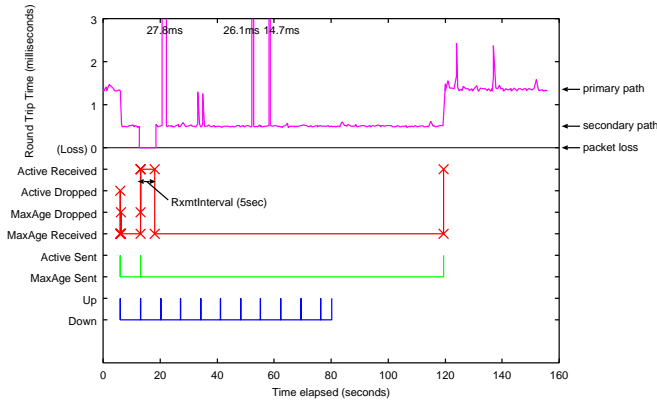
We maintain a figure of merit named *penalty* for each route which is incremented each time it flaps. This gives the degree of instability of that route. *penalty* is increased by a constant *DefaultPenalty* when a *Down* event occurs for the route. For the routes which are flapping it is their availability which is kept suppressed rather than their unavailability. When the *penalty* goes beyond a configured *Suppress* value, the route is kept from being advertised. During the period which goes without a flap, the *penalty* continues to be decayed exponentially at a rate by which the *penalty* will be reduced by half in configured *HalfLife* seconds. When the *penalty* falls below a configured *Reuse* value, the route that was suppressed is now advertised and the router originates an AS-External-LSA with age set to 0.

For this experiment we set the parameters *DefaultPenalty*, *Suppress*, *Reuse* and *HalfLife* 1000, 3000, 2000 and 20 respectively.

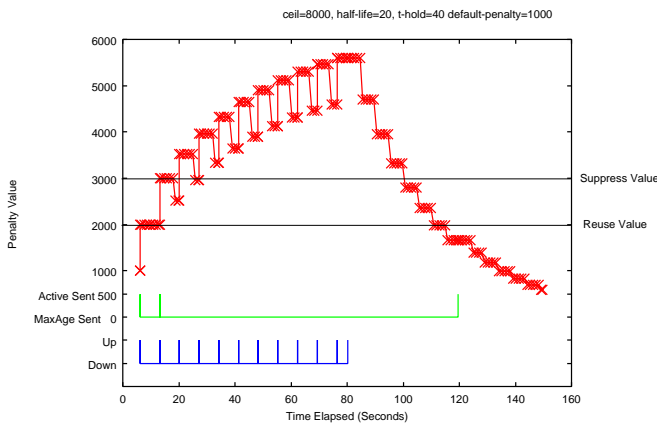
### 3.2. Experiment result

We applied this technique over the worst case scenario when there was a flap cycle of 7 seconds in which the *Up* event was for some ( $0 < t < 1$ ) seconds and the *Down* event for some 7 seconds. Our results are shown in the figure 5.

The lowest line shows how the route was flapped. It was brought *Up* for some small time and was brought *Down* for the next 7 seconds. The second line from the bottom in the chart shows the OSPF send event which has significantly improved even in case of persistent real *Up/Down* events (i.e. when the route is flapping). RTT of UDP communication (top line in the chart) shows that while damped there are no packet losses or routing loops formed and the communication peacefully uses the secondary path oblivious to the flapping which is happening. We then stopped flapping of the address at *R1* and it took approximately 20 seconds of stability for *R1* to decide advertising itself as the primary path. The UDP communication is then restored to the primary path.



**Figure 5.** RTT, OSPF send/receive event during flapping of 7 seconds cycle (*Up* immediately followed by *Down* and remains *Down* for 7 seconds) with flap damping



**Figure 6.** *penalty* transition during flapping of 7 seconds cycle (*Up* immediately followed by *Down* and remains *Down* for 7 seconds) with flap damping

The *penalty* transition is shown in figure 6. It shows the relationship between the *Up/Down* event, the *penalty* value and the OSPF behavior of originating LSAs. At each *Down* event, *penalty* is increased by 1000 (referred to as *DefaultPenalty*) while decaying is applied simultaneously. Once the *penalty* reaches a value of 3000 (*Suppress*) the damping starts. The damping feature maintains the route’s state as “*Down*” during the period when it is damped, and *RI* stops any further origination and flushing of the corresponding AS-External LSA for this prefix. If the flapping stops, *penalty* is not increased any more and it only decays with the passing time. When the *penalty* falls below 2000 (*Reuse*), it is assumed that the route is now stable and an AS-External-LSA is originated for this prefix.

In our experiment the reason why the LSA was not originated immediately after *penalty* falls below *Reuse* limit is because of the time granularity used in our implementation. It is a constant *DeltaReuse* set to 10 seconds.

It is possible to configure flap damping to handle short term severe route flaps and even milder (long term) repeated route flaps. A less aggressive suppression can be applied to the case where no alternate path exists. In the simplest case, a more aggressive suppression should be applied if any alternate route/path exists.

#### 4. Conclusion

The behavior of OSPF against flaps has been investigated in this paper. We show how flaps adversely affect OSPF routing and the communication environment in general. The current OSPF fixed timer damping function is not “flap tolerant” as each flap may lead to a few seconds of lost connectivity. Moreover *MinLSInterval* and *MinLSArrival* timers can sometimes adversely affect the network convergence. Further, as OSPF advertises and computes SPF based upon incorrect information, there are lower chances of switching to the alternative stable route even if one exists.

We implemented route flap damping to originate AS-External-LSA only when we are sure that the route redistributed into OSPF is stable. This has shown considerable improvement over cases when none of such techniques were applied.

Though it has not been verified but we expect the same solution to give similar results for the other link state routing protocol such as IS-IS[1].

#### 5. Future work

OSPF is built around links and any link/adjacency change in an area triggers full SPF. If this happens periodically then it can result in persistent SPF computations. Running SPF is a very CPU intensive activity and consumes a

lot of system resources and any unnecessary triggers should be avoided as much as possible. Thus flap damping feature should be applied to links/adjacency flaps also.

OSPF can be extended to damp

- adjacencies if a router flaps
- links coming up and going down

There is an issue when applying damping feature to damp adjacency flaps. It is called “traffic shift” which defines the movement of massive traffic from one route to another. If the adjacency flap is caused due to congestion on the link then a traffic shift will be happen. Now the original damped link will be soon stabilize because the traffic (and possibly congestion) has moved to another link, and so comes up as available. Once it starts getting back all its traffic, it will start getting congested again, and the adjacency breaks again. This can cause repeated traffic shifts with no apparent solution.

We need to do some more investigation to actually see how damping router/link flaps affects the stability and overall convergence of the routing domain.

## References

- [1] *Intermediate system to Intermediate system routing information exchange protocol for use in conjunction with the Protocol for providing the Connectionless-mode Network Service (ISO 8473)*. ISO/IEC 10589:2001.
- [2] R. Coltun, D. Ferguson, and J. Moy. *OSPF for IPv6*, December 1999. RFC 2740.
- [3] J. Moy. *OSPF Version 2*, April 1998. RFC 2328.
- [4] Y. Ohara. *Zebra OSPFv3*. URL:<http://www.sfc.wide.ad.jp/yasu/research/ospf-v3.html>.
- [5] J. Postel. *User Datagram Protocol*, August 1980. RFC 768.
- [6] T. N. Project. *The NetBSD Project*. URL:<http://www.netbsd.org>.
- [7] T. Z. Project. *GNU Zebra*. URL:<http://www.zebra.org/>.
- [8] C. Villamizar, R. Chandra, and R. Govindan. *BGP Route Flap Damping*, November 1998. RFC 2439.