Implementation of a BGP Route Flap Damping Algorithm for the Bird Routing Project

Alexandre Chappuis, Bastian Marquis and David Klopfenstein @ EPFL.ch

***Abstract*—Today's Internet stability strongly relies on the good behavior of dynamic routing protocols such as BGP (Border Gateway Protocol), that enables routing between Autonomous Systems. Route flapping is a well-known and undesirable phenomenon occuring in both commercial and private networks. In this report, we explain our implementation of the RFC 2439, BGP Route Flap Damping, for one famous Open Source routing software suite, the Bird Routing Project.**

We also present results of an experiment where we compared the number of BGP updates sent with and without route damping enabled, in a setup where only one of the bgp router was receiving advertisments for flapping routes.

I. Introduction

The inter-domain routing protocol BGP is still surviving to the gigantic growth of the Internet that started during the last decade. However, some widely used applications, such as Skype, still suffer from weaknesses of that protocol. The main problems are twofolds: Firstly, BGP has a slow convergence rate, meaning that a change at one location takes quite some time to be propagated to the other ends of the network. Secondly, if a node becomes unstable, for example if its connectivity constantly comes up and down, it will have bad consequences on the network, both in terms of useless processing at BGP routers and unnecessary routing traffic. Routes advertised and withdrawn at regular interval of time are said to be *flapping*.

Many approaches to counter route flapping have been developped in the late 90's. RFC 2439[1] standardized one such approach : it basically blocks updates from flapping routes, and does so until the routes are deemed stable again. This RFC has been used it extensively for many years, in both commercial and open source routers.

Although this standard is not recommended anymore[2] in today's routers, we wanted to implement it for the Bird Routing Project[3], hoping that it will serve as a good basis for future possible improvements and extensions to this RFC. There exist many variants of the Route Flap Damping alorithm and the community has not lost its interest in finding robust mechanisms that could allow BGP to be more resilient.

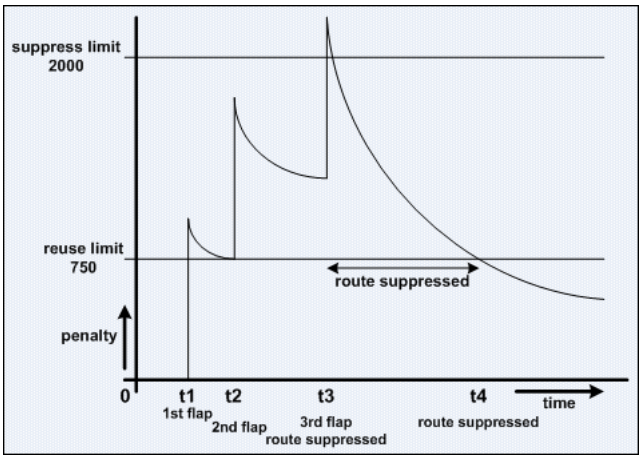
II. Overview

RFC 2349 seeks to limit the impact of route flapping by "damping" (*i.e.* ignoring packets of) missbehaving routes. The solution must be able to distinguish flapping routes from good routes, consume few resources, both in terms of memory usage and process time. The RFC solves these problems by assigning each route a penalty term. Whenever this penalty term for a given route reaches a certain threshold, further advertisements for that route are ignored. This penalty term varies over time : it is increased when the route becomes unreachable and decays as long as the route stays stable. As soon as the figure of merit goes below a *reuse threshold*, the route can be used again.

The figure of merits decays exponentially over time. Exponential decay has several advantages : it can be implemented very efficiently using precomputed *decay arrays*. Also, with exponential decay, the figure of merit keeps trace of previous instabilities for a fairly long time : old instabilities become less and less important over time, while newer ones have more weight.

Network administrators have lots of freedom in choosing the behavior of the penalty term : they can control the half-life of the penalty term, its maximum value (thus controlling the maximum time a route can be suppressed) and both the reuse and cut thresholds.

Here is an example showing how the figure of merit evolves over time (the illustration comes from [5]). The route flaps three times, exceeding the cut threshold only after the third flap. The route is reused as soon as its penalty term goes below the reuse limit.



Penalty terms are re-computed at regular time intervals using timers. Also, every time an update for a route is received, the penalty term for that route is re-computed.

The RFC proposes several optimizations to decrease processing time, at the cost of a slightly bigger memory footprint. *E.g. reuse lists* are used to not have to recompute the figure of merit of all routes : damped routes with similar penalty terms are grouped together in a same reuse list. Penalty terms of the routes in a given reuse list are then re-computed only when necessary.

Figure 1. Topology used for our simulation

Figure 2. Evolution of dampened paths for Simulation 3

III. Implementation

this part is kind of straightforward -> just explain how we did it with bird. cite coder's doc + github rep. of code

A. Data structures

B. Processing withdrawals

C. Processing route advertisements

D. Configuration parameters

E. Timers

F. Miscellaneous

IV. Evaluation

A. Overview

We used a topology of 27 routers in the NSL cluster in order to evaluate our implementation. This topology consits of 10 Autonomous systems. Router number 3 with network prefix 10.0.0.3/24 was the the one that fed the whole network 10.0.0.0/24 with updates coming from one MRT dump archive (updates.20100401.1729), that we got from the Route Views project[6]. This archive corresponds to 15 minutes of real BGP traffic that were sent by 11 BGP routers. The topology used is represented in figure x.

We ran 3 simulations and recorded every 10 seconds the total number of updates and withrawals received from all neighbors for all routers. When route damping was enabled, we also recorded the number of route dampened at any given time.

1) The first simulation was made without route damping and with the official bird daemon....
2) The seconds was made with our bird implementation and default damping parameters....
3) The third was the same as the second, except for the cut_threshold parameters that was set to 1000 instead of 1500....

B. Evolution of dampened paths

it represent time vs. the actual number of dampened routes in AS2 (actually, router 3)

also note that we aggregated results for each AS (for better readability)

C. Number of route updates

it's the difference between graph sim1 and graph sim3 -> i.e we have less updates in SIM3 ! :-)

maybe we can also display one graph with the evolution -> show that the number is quite huge !! and that's why we show the difference rather than the two graph on top of each other !

Figure 3. Difference of import updates between simulations 1 and 3

D. Number of withdrawals

works for sim3, not for sim2 !

V. Conclusion

show importance of stability

VI. Future work

possible extensions
real scale tests

VII. Acknowledgement

References

[1] The RFC 2439, BGP Route Flap Damping, http://www.ietf.org/rfc/rfc2439.txt
[2] RIPE Recommendations On Route-flap Damping, http://www.ripe.net/ripe/docs/ripe-378
[3] Bird Routing Project, http://bird.network.cz
[4] Our publicly available repository, https://github.com/alexchap/Albatros-Project
[5] itcertnotes, http://www.itcertnotes.com
[6] http://routeviews.org