Damping BGP Route Flaps*

Zhenhai Duan, Jaideep Chandrashekar, Jeffrey Krasky, Kuai Xu, Zhi-Li Zhang

Department of Computer Science and Engineering
University of Minnesota
{duan, jaideepc, jkrasky, kxu, zhzhang@cs.umn.edu}

Abstract

BGP Route Flap Damping (RFD) is anecdotally considered to be a key contributor in the stability of the global Inter-Domain Routing system. It works by suppressing advertisements about persistently flapping routes, which otherwise would be propagated throughout the Internet. However, it was recently shown that relatively stable routes, i.e., routes that only fail occasionally, can be incorrectly suppressed by this mechanism for substantially long periods of time. This phenomenon can be traced back to the complex interaction between BGP path exploration and how the RFD algorithm identifies route flaps. In this paper, we study a distinctive characteristic of BGP path exploration following a single network event such as a link or router failure. Based on this characteristic, we distinguish BGP route updates during BGP path exploration from route flaps, and propose a new BGP Route Flap Damping Algorithm, *RFD+*, with the following properties — 1) it can correctly distinguish between route flaps and path exploration; 2) it can suppress routes that are frequently and persistently changing; and 3) it does not affect routes that fail occasionally.

In addition to presenting the algorithm and the relevant properties, we also conduct simulations to illustrate the performance of our algorithm. Lastly, we present a *simplified*, *localized* route flap damping algorithm to demonstrate the possibility of *lightweight* mechanisms to improve Internet stability.

1 Introduction

Internet routing instability has an adverse impact on application performance. It manifests itself with increased network latencies and packet losses [9, 12]. There have been several measures deployed on the global Internet that have a positive impact on stability. For example, rate limiting advertisements in BGP throttles how often information is exchanged between neighbors; network prefix aggregation with *CIDR* limits the instability at the edge of the Internet from reaching the core [3]. Another mechanism to improve stability is BGP Route Flap Damping [17], which, unlike the others, was designed explicitly to limit route instability.

As stated in [17], the goals for the deployment of RFD are *1) to provide a mechanism capable of reducing router processing load caused by instability; 2) to prevent sustained route oscillation; and 3) to do so without affecting the convergence time of generally well behaved routes*. It works as follows: each router (on which RFD is enabled) maintains a penalty counter for every neighbor and prefix announced by that neighbor. This counter is incremented by a preset *penalty* when there is a route update (or withdrawal), and is exponentially decayed in the absence of any updates. If the penalty counter exceeds a configured *suppression threshold*, any route announced by the neighbor for the prefix in question is excluded from the BGP path selection process, i.e., it is *suppressed*. The penalty decays over time (in the absence of updates), and eventually falls below a configured *reuse threshold*, at which time it is re-admitted into the path selection process.

RFD is quite effective in damping *persistent and frequent* route updates, or simply route flaps [17], and it is generally considered to be one of the main contributors to the overall routing stability of the Internet [7]. At the same time, RFD sometimes incorrectly penalizes relatively stable routes. In other words, the third objective is not satisfied. In recent work by Mao *et al.* [10], it was shown that RFD can adversely affect the convergence times for routes that fail occasionally. In one particular example, it was shown that the route to a certain network can be suppressed or up-to an hour, even if it was withdrawn exactly once and re-announced soon after. This phenomenon is a result of the complex interaction between RFD and the BGP path exploration that follows a link or router failure. Intuitively, by virtue of the path vector nature of BGP, a router could potentially learn, from its neighbors, a large number of paths to a destination. In the event of the route being withdrawn at the destination, path exploration is triggered, wherein the router explores a large number of alternate paths. It might well be that the destination itself is no

*This work was supported in part by the National Science Foundation under the grants ANI-0073819, ITR-0085824, and CAREER Award NCR-9734428. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

longer reachable through any path, but this is not known at the router we are considering. This results in the router selecting a new path (because the old one was withdrawn), propagating it only to find out a little while later that the path just chosen was invalid, and so on, until all the paths have been tried and discarded. The interested reader is directed to [2, 10] for a detailed account of this problem.

To address this issue, there are two possible approaches. The first approach tries to preserve the existing RFD algorithm, but uses less aggressive damping parameters (such as smaller penalty increments, larger suppression thresholds, etc.) [2]. Clearly, this has the side effect of being more "forgiving" of actual instability, which is not desirable. The second approach would be to enhance or modify the RFD algorithm so that it can correctly distinguish path exploration from persistent route flaps, and suppress the latter only. One such attempt towards this approach is the *Selective Route Flap Damping* (SRFD) algorithm [10]. However, as will be shown later, the operation of SRFD is incorrect in certain situations. In fact, the underlying assumption about BGP path exploration upon which SRFD is based turns out to be inaccurate. Consequently, in some cases, SRFD may still suppress a relatively stable route for a long time.

It is our belief that to fundamentally address this problem we need to correctly identify the distinguishing features that set BGP path exploration apart from a route flap. We could then use these features in algorithms that would successfully distinguish path exploration from route flaps (which can then be suppressed). In this paper we first study the distinguishing features that set apart route flaps from path exploration. Based on this "signature", we propose a new BGP Route Flap Damping algorithm, RFD+, which has the following properties: (1) it correctly distinguishes between BGP updates during path exploration and route flaps; (2) it is able to suppress routes that are frequently and persistently changing; and (3) it does not affect routes that fail occasionally. In addition to presenting the algorithm and its properties, in this paper we also conduct simulations to illustrate the performance of our algorithm. Furthermore, a simpler localized route flap damping algorithm is discussed to demonstrate the possibility of employing lightweight algorithms to combat route flaps.

The remainder of the paper is structured as follows. In Section 2, we formally define BGP path explorations and route flaps. One example is also presented to demonstrate how the BGP Route Flap Damping algorithm may affect the convergence time of relatively stable routes by mistaking BGP updates during BGP path exploration as route flaps. In Section 3, we present a distinctive characteristic of BGP path explorations, which help us distinguish BGP updates during a BGP path exploration from route flaps. A new BGP route flap damping algorithm, RFD+, is studied in Section 4. Section 5 conducts simulations to compare the performance difference of SRFD and RFD+. A localized route flap damping scheme is presented in Section 6. In Section 7, we discuss related work. Section 8 concludes the paper and discusses future research work.

2 Background and Motivation

2.1 Border Gateway Protocol

We model the Internet as an undirected graph $G = (V, E)$, where V is the set of nodes, each of which represents a single Autonomous System (or AS for short), and E is the set of edges among the ASes. Although an AS may contain many BGP routers, we abuse the term *node* to refer to both an AS and a BGP router, since the exact meaning is clear from the context.

An edge exists between two ASes if and only if they exchange BGP information. Consider two nodes i and j. If there is an edge between the two nodes, we say that node i is a neighbor of node j and vice versa. Moreover, we denote the edge between node i and node j as (i, j). It is clear that if there is an edge between two nodes, there must be some underlying physical link(s) connecting them.

We now briefly describe the operation of BGP at a single router. We only present aspects of the protocol that are relevant to our discussion in this paper. For a complete description of the protocol, readers are directed to [15]. For simplicity, the following description is with respect to a single destination.

When a router receives routing information (essentially a BGP Update message for the destination), it installs the routes in a neighbor specific routing table. The set of all routes to the destination could be called the set of *candidate routes*. Subsequently, it invokes a *path selection process* to determine which of the *candidate routes* it will use — which we can term the *best path*. The path selection is based upon a locally configured *policy*. To keep the discussion simple, we assume that a local policy assigns a high preference to shorter paths (using the smallest router id to break a tie), unless otherwise specified.

Once a best path is selected, the router sends this route to its neighbors using BGP UPDATE messages[1]. A BGP UPDATE message announces a path that is potentially valid or it withdraws an existing route. In the second case, the recipient is instructed to remove the route learned earlier from the sender.

To constrain the amount of BGP routing traffic exchanged, a $minRouteAdvertisementInterval$ (or *MRAI*) timer is used to throttle announcements, requiring that *MRAI* seconds elapse between successive route announcements. Note that this timer only applies to route announcements; route withdrawals are immediately propagated to prevent the black holing of traffic (where a node forwards traffic for an invalid route, which is subsequently dropped further along the path).

[1]Excluding the neighbor that the router learns this particular route from.

2.2 BGP Path Explorations and Route Flap Damping

Here we discuss the interaction between BGP path exploration and the BGP Route Flap Damping algorithm (RFD) [17], and demonstrate how a single route flap can cause routes to be suppressed for a relatively long time. To aid our description, we first need to clearly define some notation.

2.2.1 Network Events and BGP Events

For clarity, we distinguish between *network events and BGP events*. Network events are defined as *original network* dynamics such as link/router failures and recoveries that trigger the generation of BGP update messages. For simplicity, we also refer to policy changes or policy disputes that trigger BGP route changes as network events [6][2]. We further classify network events into failure events or recovery events — depending on their effect upon the BGP routing protocol. In response to a network failure event, a BGP speaker may send out a withdrawal or select a less preferred route (if the more preferred routes have been withdrawn). On the other hand, following a network recovery event, better (more preferred) routes become available at a router and are announced to its neighbors. Examples of network failure events include link failure, router failure, and policy-related route withdrawal. Link and router recoveries, as well as policy-related route re-announcements are instances of network recovery events.

BGP events are triggered by network events, or recursively by other BGP events announced by BGP update messages. Intuitively, BGP events are simple messages (announcements or withdrawals) being generated (or propagated). We will abstractly denote a BGP event as ϕ, which can be either a route announcement or a route withdrawal. In the former case, we abuse notation and also use ϕ to refer to the actual path contained in the announcement.

2.2.2 BGP Path Exploration, Route Flap, and Persistent Route Flap

By virtue of the path vector nature of BGP, a router could potentially learn a large number of paths to a destination from its neighbors. Let us consider an event that causes the destination to become disconnected from the rest of the Internet. The exact location of the event (or the nature of the event) is not carried in the BGP events that are triggered. Consequently, when routers receive a withdrawal, they simply switch to a path with a lower preference — which is in turn announced to their neighbors. However, since there is really no valid path to the destination, each of these less preferred paths is withdrawn eventually, and the cycle continues until all of the paths are withdrawn from the system. This phenomenon can be termed *BGP path exploration*, and is an inherent artifact of all path vector protocols[3].

To define BGP path exploration formally, consider an arbitrary node i in the network. BGP path exploration, in the context of node i, is simply the sequence of BGP events **generated** by the node following a single network failure event. At the end of the path exploration, the node reaches a new stable state, i.e., it does not generate any more BGP events (if we can assume that no other network event takes place in this time). Similarly, when a failure is repaired, nodes can explore a number of paths before settling on a stable path. Thus, we distinguish between failure path exploration (which is a result of a network failure event) and recovery path exploration (in response to a network recovery event). In the rest of the paper, we do not explicitly prefix the type (failure or recovery) when mentioning path explorations except when it is not clear from the context.

Given the potentially large number of transient BGP updates generated by a node during path exploration, it is possible that one of its neighbors may decide that the routes being announced by the node are not stable. In the next subsection, we demonstrate the interaction between path exploration and RFD.

A route flap could be defined as the BGP event sequence that is associated with a network failure event and a corresponding network recovery event (occurring soon after). Let ϱ denote a route flap and $|\varrho \in T|$ the number of occurrences of the route flap during a given time interval T. Let τ be a configurable constant parameter. Then if $|\varrho \in T| \geq \tau$, we say that ϱ is a *persistent* route flap.

2.2.3 BGP Route Flap Damping and its Interaction with BGP Path Explorations

Persistent route flaps increase Internet routing traffic and degrade Internet performance. The objective of BGP Route Flap Damping (RFD) is to suppress the usage and spread of such persistently flapping routes *without affecting the convergence time of relatively stable routes* [17]. As mentioned earlier, RFD is a penalty-based scheme. For every neighbor, node i maintains a penalty counter for each network prefix, which is increased by a preset penaly whenever a BGP update is received from the neighbor (regarding the network prefix). When the counter exceeds a pre-defined suppression threshold, all the related routes from the neighbor (routes to the particular destination prefix announced by this neighrbor) are excluded from the BGP path selection process, or to put it in another way, they are suppressed. The penalty counter decays exponentially over time, and when it is below a reuse threshold the

[2]We could take the stand that such policy based changes are initiated by some physical event.

[3]In this, we are deviating from the common practice of terming any sequence of path changes as *path explorations*, irrespective of the underlying cause.

corresponding routes can participate in the BGP path selection process again. The penalty counter decays as follows: Let $\Pi(t)$ denote the penalty counter at time t, then for $t' \geq t$

$$\Pi(t') = \Pi(t)e^{-\lambda(t'-t)}, \tag{1}$$

where λ is a system parameter, which is normally configured through a *Half-life* parameter H by the equation $e^{-\lambda H} = 0.5$.

Even though RFD is effective in damping persistent route flaps, it was recently shown that RFD may suppress a relatively stable route for a long time. To understand this better, in the following description we present a real sequence of BGP advertisements that demonstrate the problem. The advertisements shown in Table 1 are for a single prefix 198.133.206.0/24 on January 19, 2003. This prefix was selected because it is used by the BGP Beacons project, where a set of prefixes are announced and withdrawn at well defined intervals[14]. The BGP updates were collected at the University of Minnesota network. The first column of the table is the time at which the BGP message was received at the observation host[4]. In the second column, a path indicates that the advertisement was an announcement, whereas the absence of a path indicates a withdrawal advertisement. The third column represents the value of the penalty counter at the time the advertisement was received, while the fourth column represents the value after the advertisement has been processed (and the penalty $\mathcal{P}$ added). In order to compute the penalties, we use the default Cisco RFD parameters (see Table 2). Thus, the first three advertisements incur a penalty of 500 each (since they correspond to updates), while the last one incurs a penalty of 1000.

Notice that at time 13:03:48, the penalty value is greater than the *suppression threshold*, which causes the prefix to be suppressed. If we can extrapolate a little and replace the prefix by a normal prefix that for some reason failed and then came back on soon after, we would see an announcement for this a little while after the last withdrawal. In this case, since the penalty value is greater than the suppression threshold, *this announcement will be ignored* even though it corresponds to a valid repair event, and will not be considered until the penalty value decays to a value below the re-use threshold, which takes approximately 25 minutes in this example!

This phenomenon can be traced back to the complex interaction between BGP path exploration and the RFD algorithm. Recall that during BGP path exploration, a large number of BGP route updates can be advertised from a node to its neighbors. From the neighbor's point of view, the routes going through the node appear unstable and are thus suppressed by RFD *even though all the BGP updates are part of BGP path exploration*.

[4]For simplicity, we can assume that this prefix was never seen before, so we can justify $\Pi(t) = 0$.

Table 1: Interaction between RFD and BGP path exploration

Time	Path	$\Pi(t)$	$\Pi(t) + \mathcal{P}$
13:00:33	217 57 3908 1 3130 3927	0	500
13:01:00	217 57 1 2914 3130 3927	489.710	989.710
13:01:28	217 57 3908 3356 2914 3130 3927	968.596	1468.596
13:03:48	-	1318.486	2318.486

Table 2: Default Cisco RFD configuration values

Parameter	Value
Withdrawal penalty	1000
Attributes change penalty	500
Suppression threshold	2000
Half-life (min)	15
Reuse threshold	750
Max suppress time (min)	60

2.3 Selective Route Flap Damping

As discussed above, the current BGP Route Flap Damping algorithm may incorrectly delay the convergence of relatively stable routes. To fundamentally address this issue, we need to identify the essential distinction(s) between BGP path exploration and route flaps, upon which we can base algorithms that can correctly suppress persistently flapping routes while being insensitive to BGP updates during BGP path exploration. *Selective Route Flap Damping* (SRFD) is perhaps the first attempt in this direction [10]. SRFD is based on the simple observation that during a BGP path exploration, the route with the highest preference among the *current available routes* is chosen as the best route. Therefore, the preferences of the announced best routes during BGP path exploration *should* be monotonic. It is important to note that we are referring to the preference at the neighbor.

Based on this assumption, SRFD treats a sequence of routes with alternating relative preference as an indication of a route-flap. Relative preference of routes at a neighbor is defined as the comparative value of two consecutive route announcements[5]. See [10] for more details on SRFD.

SRFD was verified as correctly detecting route flaps while being insensitive to path exploration for the network configurations studied in [10]. However, the assumption about monotonic relative preference is inaccurate and consequently, in some cases, SRFD might fail to correctly distinguish between path exploration and route flaps, leading to the suppression of a well behaved route. To see why the assumption about monotonic preference changes is not true, note that when a current best route is withdrawn, a BGP speaker selects a new best route from the set of *currently available* alternative paths. However, because of topological dependencies and delays in BGP message processing and propagation, the set of *currently available* alternative paths at the router can be different at different times. Therefore, routes with alternate relative prefer-

[5]In case announcements are interleaved with withdrawals, relative preference is not well defined. In this case, SRFD looks for interleaved advertisements with the same relative preference as an indicator of a route flap.

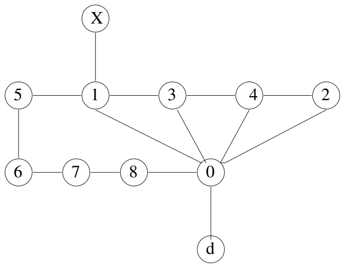
Figure 1: *fork* network.

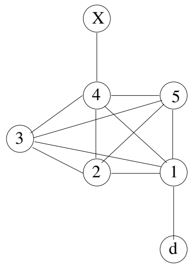
Figure 2: *clique(5)* network.

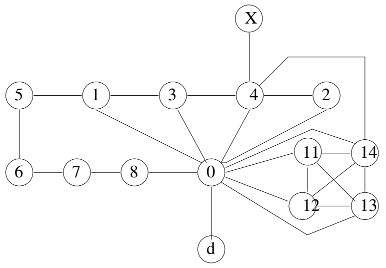
Figure 3: *fork-clique(4)* network.

Table 3: BGP updates with non-monotonic preference changes.

Stage	Routing tables	New messages	Preference
0	1(*0d, 30d, 56780d) 3(*0d, 10d, 40d) 4(*0d, 20d, 30d) 2(*0d, 40d)	- (steady state)	
	edge (0,d) is down	0→{1, 2, 3, 4, 8}W	
1	1(-, *30d, 56780d) 3(-, *10d, 40d) 4(-, *20d, 30d) 2(-, *40d)	**1→{x,3}[130d]**, 3→{1, 4}[310d], 4→{2, 3}[420d], 2→{4}W	↓
2	1(-, -, *56780d) 3(-, -, *420d) 4(-,-,*310d) 2(-,-)	**1→{x}[156780d]**, 3→{1,4}[3420d], 4→{3}W	↓
3	1(-, *3420d, 56780d) 3(-,-,-) 4(-,-,-) 2(-,-)	**1→{x}[13420d]**, 3→{1}W	↑
4	1(-,-,*56780d) 3(-,-,-) 4(-,-,-) 2(-,-)	**1→{x}[156780d]**	↓

ences may be announced by the router to its neighbors during BGP path exploration if it turns out that a "better path" than the one currently chosen happens to become available (during path exploration).

To have a more intuitive appreciation of the dynamic complexity during BGP path exploration, below we present a simple example to demonstrate that a node may announce routes with alternate relative preferences during BGP path exploration. See Appendix A for an example showing that a node may also announce a route withdrawal between two BGP route announcements with the same preference during BGP path exploration.

For simplicity, we adopt the following discrete-time synchronized BGP model. In each discrete-time stage, a node processes *all* the pending announcement messages received in the last stage. After processing these messages, the node may update its neighbors accordingly. If the best route to a destination prefix is changed, the node sends the new best route to its neighbors. If the network prefix becomes unreachable, a BGP withdrawal message is sent. After all the nodes finish this processing, the system advances to the next stage. Note that, in each stage at most one update message (either an announcement or a withdrawal) is sent from a node to each of its neighbors.

Figure 1 presents a simple AS-level network topology; we refer to it as the *fork* network. The numbers or letters in the figure denote the id of the corresponding nodes. For simplicity, we only consider one destination node d and all the routes are given with respected to this node. We assume that node 5 prefers the routes announced by node 6 over those by node 1. All other nodes do not have such a local preference differentiation. Assume initially that node 0 announces a route to all of its neighbors, and this information is propagated in the network. After all the nodes enter a steady state regarding the route to node d, edge $(0, d)$ is down. Table 3 presents the subsequent BGP updates sent from node 1 to node x, in a format similar to [10]. The table has four columns. The column marked with *Stage* records the stage indexes. The *Routing Table* column presents the routes known by the nodes (for clarity the table only shows the routing tables for nodes 1, 2, 3, and 4). As an example $1(*0d, 30d, 56780d)$ indicates that node 1 has three routes to node d by going through node 0, 3, and 5, respectively; the route marked with an asterisk ($0d$ in this example) is the best route chosen by the node. A dash sign indicates an invalid route. The third column, *New messages*, provides the new messages (announcements of a new route, or withdrawals) generated by the nodes. These messages are processed in the next step. New messages are given in the following format: $i \rightarrow \{j_1, j_2, \ldots, j_k\}[path]$, where i is the originator of the message, j_1 to j_k are i's neighbors to which node i advertises the new route $path$; if $[path] = W$, the announcement is a path withdrawal. To simplify the description of the example, we assume the (processing and propagation) delay on the path from node 8 to 5 through nodes 7 and 6 is sufficiently large, so that node 5 has not withdrawn the route to node d at the last stage in the table. The last column gives the changes in the preferences of the routes announced by node 1 to node X, where ↑ indicates an increase in route preference, whereas ↓ a decrease. From the table, we see that routes with alternate preferences can indeed be announced during path exploration.

3 Characterizing BGP Path Exploration and Route Flap

In this section we present a simple yet unique characteristic of BGP path exploration. Based on this *provable* property of path exploration, we can correctly distinguish BGP path exploration from route flaps. This forms the basis for the next section, where we design a correct BGP route flap damping algorithm.

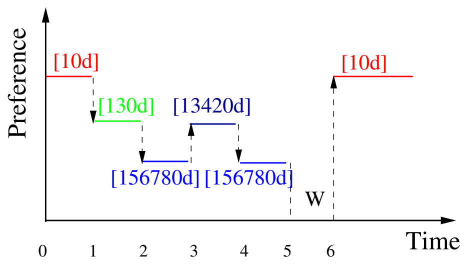

Figure 4: Route updates

Before we present the main result of this section, let's examine the example in Section 2.3 more closely. Without loss of generality, let's assume that at stage 5, node 1 withdraws the route $[156780d]$ from node x, and at stage 6, node 1 re-advertises the route $[10d]$ (assuming edge $(0, d)$ comes back sometime before stage 6). Figure 4 presents the route updates sent to node x from node 1, as well as the preferences of the routes *at node* 1. Note that in the figure, the absolute value of the preference of a route is not important, we are more interested in the relative preference of two routes, as indicated by the arrows in the figure. A downward arrow indicates a decrease in the preference, while an upward arrow indicates an increase in the preference. From the figure, we see that during the BGP failure path exploration (before stage 6), route $[156780d]$ is advertised by node 1 twice at stage 2 and 4, respectively. On the other hand, routes $[130d]$ and $[13420d]$ are only advertised once. As soon as they are (implicitly) withdrawn, node 1 would not announce them again during the course of the path exploration. We note that route $[156780d]$ differs from routes $[130d]$ and $[13420d]$ in that route $[156780d]$ has the lowest preference compared to its prior and succeeding routes (ignoring withdrawals for this matter), while routes $[130d]$ and $[13420d]$ do not. Put in another way, during the course of BGP (failure) path exploration, once a route with a higher preference is replaced by a route with a lower preference, the route with a higher preference will not be advertised by the node again. Therefore, *the neighbors of the node would only see the routes with higher preferences once in a BGP path exploration*. On the other hand, in a route flap, *a route with a higher preference may be seen by the neighbors twice*. This observation could be used to distinguish a BGP path exploration from a route flap. We first state this observation as a formal proposition and present a proof. For ease of exposition, we will use P_r denote the preference of a route r, and $P_{r_1} < P_{r_2}$ to indicate that route r_1 has a lower preference compared with route r_2. To further simplify things, we assume that a BGP explicit withdrawal has the lowest preference, that is, $P_W < P_r$, for any route r.

Proposition 1 *Consider a node i and let node j be a neighbor of node i. Let Φ denote a sequence of BGP events sent by node j to node i. Without loss of generality, let $\Phi = \phi_1\phi_2\phi_3 \ldots \phi_n$, where ϕ_l is a BGP event, which can be either a BGP route announcement or an explicit withdrawal, for $l = 1, 2, 3, \ldots, n$. If Φ is a path exploration (PE), Φ must not contain the following BGP event pattern: $P_{\phi_{m-1}} < P_{\phi_m}$ and ϕ_m is a* repeated BGP route announcement. *More formally,*

$$\begin{aligned} \Phi \textit{ is PE} \quad \Rightarrow \quad & \neg(\exists m, k : 1 < m \leq n, \leq k < m - 1; \\ & P_{\phi_{m-1}} < P_{\phi_m} \;\&\; \phi_m = \phi_k) \end{aligned} \tag{2}$$

Proof: We prove this proposition by contradiction. Assume that for some m and k, the BGP event pattern indeed occurs, i.e., $P_{\phi_{m-1}} < P_{\phi_m}$ and $\phi_m = \phi_k$. Let k' be the largest of such k's, i.e., $\phi_l \neq \phi_m$ for $l = k' + 1, k' + 2, \ldots, m - 1$. Let $r_{k',m}$ be the route associated with (carried in) $\phi_{k'}$ and ϕ_m. We focus on the (sub)sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2} \ldots \phi_m$. We consider two cases.

CASE 1: Φ is a BGP failure path exploration. First let's assume that there is no BGP withdrawal in the sequence. Let l be the smallest index between k' and $m-1$, such that $P_{\phi_l} < P_{\phi_m}$. Given that $P_{\phi_{m-1}} < P_{\phi_m}$, such a route ϕ_l always exists. Now let's consider the time t when node j announces route ϕ_l to node i. It is easy to see that at time t, $r_{k',m}$ must not be available at node j. Otherwise, node j would rather announce $r_{k',m}$ to node i instead of ϕ_l. The fact that $r_{k',m}$ is not available at node j at time t (but available at node j at an earlier time, note $\phi_{k'}$) can be caused by a local network event at node j or a network event at a downstream node between node j and the destination network along the route $r_{k',m}$. Without loss of generality, let's assume the network event occurs at node f between the destination network and node j along the route $r_{k',m}$. Let t' denote the time when the network event happens at node f, where $t' \leq t$. This is also the time when node f withdraws the route $r_{k',m}$ from the upstream nodes. Notice that route $r_{k',m}$ becomes available again at node j at a time $t'' > t$ (note ϕ_m), then node f must re-announce the route at a time between $(t', t'']$. Therefore we know that the failure associated with the network event at node f must have been recovered at the time. Given that a network failure and recovery event pair is associated with the sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2} \ldots \phi_m$, we know that Φ cannot just be part of a path exploration. Therefore, we reach a contradiction.

Now let's consider the situation where at least one BGP withdrawal is contained in the sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2}\ldots\phi_m$, and let ϕ_l be the first withdrawal following $\phi_{k'}$. Consider two cases: First assume at least one of the routes $\phi_{k'+1}, \phi_{k'+2}, \ldots, \phi_{l-1}$ has a lower preference compared to $r_{k',m}$. Then following the same argument as above, we can show that a network failure and recovery event pair is associated with the sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2}\ldots\phi_m$, and again we reach a contradiction. Now assume all the routes $\phi_{k'+1}, \phi_{k'+2}, \ldots, \phi_{l-1}$ have a higher preference compared to $r_{k',m}$. Let t denote the time when the withdrawal ϕ_l is sent from node j to node i. Given a withdrawal is sent at time t from node j, we know that route $r_{k',m}$ is not available at node j. By noting that route $r_{k',m}$ is later announced to node i by node j (ϕ_m) and following the same argument as above, we see that a network failure and recovery event pair is associated with the sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2}\ldots\phi_m$, and Φ cannot just be part of a path exploration. We reach a contradiction again.

CASE 2:Φ is a BGP recovery path exploration. First let's assume that there is no BGP withdrawal in the sequence. Let l be the smallest index between k' and $m-1$ such that $P_{\phi_l} < P_{\phi_m}$. Given that $P_{\phi_{m-1}} < P_{\phi_m}$, such a route ϕ_l always exists. Now let's consider the time t when node j announces route ϕ_l to node i. It is easy to see that at time t, $r_{k',m}$ must not be available at node j. Otherwise, node j would rather announce $r_{k',m}$ to node i instead of ϕ_l. Put in another way, at time t, route $r_{k',m}$ is replaced by some less preferred routes at node j. However, during a BGP recovery path exploration, once a route is present at a node, it can only be replaced by a route with an increased preference. We reach a contradiction. The situation where at least one BGP withdrawal is contained in the sequence $\phi_{k'}\phi_{k'+1}\phi_{k'+2}\ldots\phi_m$ can be proved in a similar manner, i.e., leading to a contradiction. We omit it here.

Combining the above two cases, we have

$$\Phi \text{ is PE} \quad \Rightarrow \quad \neg(\exists m, k : 1 < m \leq n, 1 \leq k < m-1; \\ P_{\phi_{m-1}} < P_{\phi_m} \;\&\; \phi_m = \phi_k) \tag{3}$$

■

It is worth noting that the condition $P_{\phi_{m-1}} < P_{\phi_m}$ is crucial. In both BGP path explorations and route flaps, the same route can be advertised repeatedly by a node to its neighbor (See Figure 4 where route [156780d] is announced twice during the BGP path exploration). However, during the course of a BGP path exploration, the repeated route must have a lower preference compared to the adjacent routes announced.

Proposition 1 provides an essential property of the BGP event sequence of a BGP path exploration. To facilitate its usage, below we present its contrapositive as a corollary. Using the same notation as in Proposition 1, we have

```
Input: $\Phi = \phi_1\phi_2\phi_3\ldots\phi_n$;
Output: Type of the sequence;
for ($k \leftarrow 2; k \leq n; k++$)
  if ($P_{\phi_{k-1}} < P_{\phi_k}$)
    for ($l \leftarrow 1; l < k; l++$)
      if ($\phi_l = \phi_k$)
        return ($\Phi$ contains route flap)
return ($\Phi$ is a BGP path exploration)
```

Figure 5: Classification of BGP event sequences.

Corollary 2

$$(\exists m, k : 1 < m \leq n, 1 \leq k < m-1; \\ P_{\phi_{m-1}} < P_{\phi_m} \;\&\; \phi_m = \phi_k), \quad \Rightarrow \quad \neg(\Phi \textit{ is PE}) \tag{4}$$

We assume that for a given sequence of BGP events Φ, if it is not a BGP path exploration, it must contain at least one route flap. Therefore, Corollary 2 provides us with a way to identify a route flap. Based on Corollary 2, Figure 5 presents a simple algorithm to determine if a *given BGP event sequence* contains a route flap. Essentially, if a BGP sequence contains a *repeated route with* $RP = 1$, the algorithm claims the existence of a route flap. Otherwise, it is a sequence of BGP updates during BGP path exploration. In the next section, we will present a new BGP route flap damping algorithm using this corollary. We will see how route flaps can be detected *online without mistaking BGP updates during path exploration as route flaps*.

4 $RFD+$: A New BGP Route Flap Damping Algorithm

In this section we design a new BGP route flap damping algorithm called RFD+ to damp persistent route flaps based on Proposition 1. It is able to correctly distinguish BGP path explorations from BGP route flaps, and only suppresses *persistent* route flaps. RFD+ has two components. The first one is a mechanism to identify route flaps (based on Proposition 1), and the second one is a suppressing mechanism to determine when a route should be suppressed. For the second component, we present a window-based counting scheme to suppress persistent route flaps. However, it should be emphasized that exact nature of the suppressing mechanism is not important. What is critical is the *correctness* of the scheme to identify route flaps. Indeed, other suppressing schemes, such as using fixed timers (suppress for a fixed time), the penalty-based exponentially decaying scheme used in the current BGP Route Flap Damping algorithm [17] can as well be employed in RFD+, since this does not affect how route flaps are detected, but merely specify what is to be done, once they are detected. For simplicity, all the following discussions are made with respect to a destination network d.

First, let's define some notation. Node i classifies a neighbor j into two states: *suppressed or eligible*. If neighbor j is suppressed (or simply (j, d) is suppressed), routes announced from j are excluded from the BGP route decision process. Routes from neighbor j can participate in the BGP route decision process at node i only if node j is eligible (or simply (j, d) is eligible). All the neighbors of node i are initially considered eligible.

4.1 Relative Preference Community Attribute

Note that in Corollary 2, it is required that when node i receives a new route from neighbor j, it must know the relative preference of the new route compared to the previous route *at node* j. For this purpose, we introduce a new Community Attribute called Relative Preference (RP) (similar to SRFD [10]). When node j advertises a route to its neighbor i, it inserts the RP community attribute in the update message. This RP attribute indicates the relative preference of the new route compared to the previous one at node j. RP is an one-bit community attribute. It is set to 1 if the new route has a higher preference. Otherwise $RP = 0$. If the RP attribute is absent in the update message, the receiving node will take the default value of RP, which is 0.

4.2 Route Flap Identification

Let R_j^d denote a data structure for maintaining the routes announced from node j. For ease of later discussions, let $r \in R_j^d$ denote the fact that r is in the data structure, and $r \rightarrow R_j^d$ the insertion of route r into R_j^d (note that the real insertion only occurs if $r \notin R_j^d$).

Now consider that the current best route announced by node j is replaced by a new route r. If $r \notin R_j^d$, then $r \rightarrow R_j^d$. Otherwise, if $r \in R_j^d$ and the carried $RP = 1$, node i knows that a route with an increased preference is repeated. Based on Corollary 2, node i knows that a route flap has occurred. At this time, all the routes in R_j^d are cleared, i.e., $R_j^d = \emptyset$, for the following reasons: First, note that all the routes in R_j^d will be repeated in the next course of the route flap. If node i does not remove the routes in R_j^d, the repetitions of all the routes may be counted as route flaps, which is not correct. Second, the first repeated route with $RP = 1$ may not be the most preferred route at node j. A less preferred route may first appear at node j in the corresponding BGP recovery path exploration, and then later it could be replaced by more preferred routes. Node i will over-count the route flaps if the other (more preferred) routes are not removed.

```
0.   $j$: neighbor of node $i$; $d$: destination;
1.   Upon receiving an route $r$ from $j$:
2.     if ($r \notin R_j^d$)
3.       $r \rightarrow R_j^d$;
4.     else if ($r \in R_j^d$ and $RP = 1$)
5.       /* a route flap is identified */
6.       $\eta_i^d + +$;
7.       $R_j^d \leftarrow \emptyset$;
8.   At the end of each time window $T$:
9.     $\Lambda_j^d \leftarrow \alpha\Lambda_j^d + (1 - \alpha)\eta_j^d$;
10.    $\eta_j^d \leftarrow 0$;
11.    if ($\Lambda_j^d \geq \tau$ and (j,d) is eligible)
12.      suppressing (j,d);
13.    else if ($\Lambda_j^d < \upsilon$ and (j,d) is suppressed)
14.      (j,d) $\leftarrow$ eligible;
```

Figure 6: Pseudo code of RFD+.

4.3 Persistent Route Flaps Suppression

We use a window-based counting scheme to identify persistent route flaps. Let T denote a configurable time interval (window). Let τ and υ be two configurable constants, where $\upsilon \leq \tau$. We refer to them as suppression threshold and reuse threshold, respectively. We will see their usages shortly. To track the number of route flaps, node i maintains a counter η_j^d for each neighbor j. At the beginning of each time window T, η_j^d is set to 0. Whenever a route flap from neighbor j is identified by node i, η_j^d is advanced by one. At the end of each time window, η_j^d contains the number of route flaps that occurred in the last time window.

We could *immediately* suppress the routes announced by neighbor j if $\eta_j^d \geq \tau$. However, a more graceful way would be to rely on the long-term trend of route flapping dynamics instead of what happens in one time window. Let Λ_j^d denote the average number of route flaps in the current window and previous windows. Λ_j^d is computed using exponential-weighted moving average (EWMA), i.e.,

$$\Lambda_j^d \leftarrow \alpha\Lambda_j^d + (1 - \alpha)\eta_j^d,$$

where α is a configurable parameter used to control the contribution of the route flaps history to the calculation of Λ_j^d. Λ_j^d is initialized to 0 when the system first starts. At the end of each time interval, Λ_j^d is re-computed. If $\Lambda_j^d \geq \tau$, the related routes are suppressed. On the other hand, if $\Lambda_j^d < \upsilon$, the related routes become eligible again.

Figure 6 summarizes the RFD+ algorithm.

4.4 Properties of RFD+

In this section we briefly discuss some properties of the RFD+ algorithm. From the Section 4.2 (see also Lines 4–5 in Figure 6), we know that RFD+ only claims a route flap if a route is repeated with increased preference (RP = 1). On the other hand, from Corollary 2, we know that any BGP event sequence containing a repeated route with RP = 1 cannot just be part of a BGP path exploration. Therefore,

Remark 1 *RFD+ distinguishes route flaps from BGP path explorations, and any BGP events of a BGP path exploration will not be wrongly taken as route flaps.*

Now let's turn our attention to persistent route flaps. First we assume that no two route flaps are interleaved with each other. As we discussed above, when the first repeated route with RP = 1 reaches a node i from neigbor j, node i identifies this as a route flap and clears the records of the stored routes in R_j^d. The next route flap will be identified by node i in the same way, i.e., RFD+ will count every route flap once and only once. As a result, RFD+ can identify all such route flaps.

Now consider the case where the BGP event sequences of multiple route flaps interleave with each other. Recall that when node i gets the first repeated route from neighbor j with RP=1, it identifies a route flap and removes all the routes from R_j^d. Note that, even though such an operation may remove the record of other route flaps, RFD+ can always identify at least one of the (most frequent) route flaps. As long as one persistent route flap is identified, all the related route flaps will be suppressed.

Remark 2 *RFD+ can suppress all the persistent route flaps.*

Consider a node i. Note that during the course of a *single* route flap at a neighbor j, RFD+ at node i only advances the route flap counter η_j^d by *one*. This is performed when RFD+ detects the first repeated route with RP = 1. This route can be the original most preferred route before the network failure event, or an alternative path depending on which one is first avaiable at the neighboring BGP speaker j during the BGP recovery path exploration. If the most preferred route reaches node j first, there will be no more BGP route updates from node j, and the route flap is over. So in this case η_j^d is only advanced by one. On the other hand, if a less preferred route comes to node j first, it may be later replaced by other (more preferred) route. As a result, new BGP update messages will be sent by node j to node i. However, from the description of the RFD+ algorithm (Section 4.2) and also Lines 4–7, we note that RFD+ at node i has cleared the records of all the routes (particularly more preferred ones). Therefore, RFD+ will not count new BGP route updates during the recovery path exploration of the same route flap as additional route flaps, given that the conditions $r \in R_j^d$ and $RP = 1$ will not hold. Given that RFD+ only advances η_j^d once during a route flap, it will not overcount occassional route flaps as persistent ones. Therefore we have,

Remark 3 *RFD+ will not mistake occational route flaps as persistent route flaps. Therefore, RFD+ will not suppress relatively stable routes.*

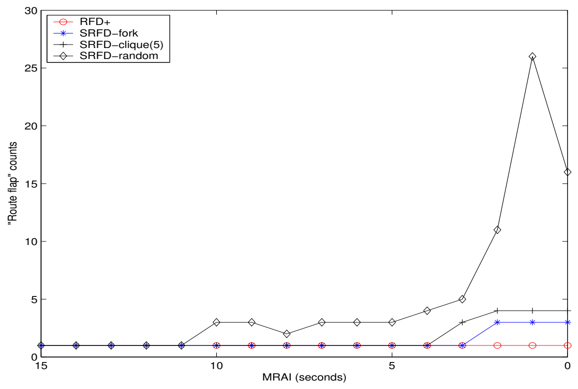

Figure 7: Comparison of SRFD and RFD+.

5 Simulation Studies

In this section we conduct simulation studies to compare the performance difference between SRFD and RFD+. For each simulation we schedule a *single* link failure at a certain time and a corresponding recovery at a later time, i.e., a single route flap. We compare the *number of route flaps claimed* by both SRFD and RFD+ from the same observation point. The discrepancy between the number of *real route flaps* occurred in a network and the number of *route flaps claimed by a route flap damping algorithm* reflects to what degree the damping algorithm mistakes BGP route updates in a BGP path exploration as route flaps. Therefore, it is a good performance indicator of route flap damping schemes in terms of *correctly* identifying route flaps.

We first describe the network configurations in the simulations. Three different network topologies are used. They are the *fork* network (Figure 1), the *clique(5)* network (Figure 2), and a network topology created by a random network generator, which we refer to as the *random* network. In all the simulations, we focus on the BGP update messages regarding a single network destination (node d in Figures 1 and 2) at a single observation node (node x in Figures 1 and 2). For the third topology, the destination and observation nodes are connected to different randomly selected routers in the random network. The resulting topology is kept consistent when studying both SRFD and RFD+.

In the *fork* network, all the edges have a propagation delay of 1 second except edge $(8, 0)$, which has a propagation delay of 3 seconds. This is to make the propagation time of BGP messages on path (5, 6, 7, 8, 0) sufficiently greater than those on other paths. (As noted in [8], the propagation delays of most BGP messages between two peers (neighbors) on the Internet are within several seconds.) Similarly, in the *clique(5)* network, all the edges have a propagation delay of 1 second except edge $(3, 5)$, which has a propagation delay of 3 seconds.

There are a total of 24 routers in the random network (including the destination and observation nodes). As mentioned above, both the destination and observation nodes have a degree of 1 (because they are attached later). All other routers have a degree between 4 and 6 (inclusive). All the edges have a propagation delay of 1 second.

The edge experiencing a single failure and recovery is the edge between the destination node and its neighbor (edge $(0, d)$ in the *fork* network and edge $(1, d)$ in the *clique(5)* network).

Figure 7 presents the *number of route flaps claimed* by both SRFD and RFD+ as a function of minRouteAdvertisementInterval (MRAI). From the figure we see that RFD+ can always correctly identify the single route flap, independent of the value of MRAI. On the other hand, for SRFD to detect the route flap correctly without over-counting, the MRAI value needs to be sufficiently large. When the MRAI value is small, SRFD mistakes some BGP route updates during path exploration as route flaps. That is, SRFD cannot independently and correctly detect the number of *real route flaps*. More specifically, consider the simulations with the *clique(5)* network. We can see that when the MRAI value drops to 3 seconds, SRFD claims there are 3 route flaps, and when the MRAI value further drops to 2 seconds or smaller, 4 route flaps are claimed by SRFD, even though there is only a single link failure and recovery in the network. Similar behavior is observed on the *fork* and *random* networks with the SRFD damping algorithm. However, it is important to note that, as demonstrated by the simulation results with the *random* network, the interaction between SRFD and MRAI can be rather complex. The number of route flaps claimed by SRFD does not necessarily decrease when the MRAI value is increased, even though it holds as a macroscopic trend. The number of route flaps claimed by SRFD depends on both the value of MRAI (which will in general reduce the number of BGP updates, see also [5]) and the ensuing BGP route update announcement pattern.

As shown by Griffin and Premore [5], for each network, there exists an optimal value of MRAI to minimize the BGP routing convergence time. However, there are no general rules to derive the optimal MRAI value and it varies from one network to another. Moreover, it is not clear if the optimal MRAI value is large enough for SRFD to correctly detect the number of route flaps. Currently, the default value of MRAI used by Internet routers is 30 seconds (which is somewhat arbitrarily chosen). However, even with $MRAI = 30$ seconds, SRFD may not be able to correctly detect the number of route flaps, as demonstrated by the following simulation conducted on the *fork-clique(4)* network (Figure 3). In this simulation, all the edges in the network have a propagation delay of 1 second. Again, the edge $(0, d)$ fails and recovers once, i.e., there is only one route flap in the network. However, at node x SRFD claims there are 2 route flaps. On the other hand, RFD+ correctly detects that there is only one route flap.

6 Localized Route Flap Damping?

RFD+ allows for incremental deployments, in particular, RFD+ confines the effects of persistent route flaps to nodes that have not deployed the algorithm (and their neighbors). On the other hand, RFD+ is relatively expensive. In RFD+, BGP route updates need to carry the new Relative Preference community attribute, and nodes need to maintain tables regarding each destination for each neighbor. In this section, we discuss a simple *localized* BGP route flap damping algorithm, called *LRFD*, where a node *only* damps persistent route flaps *if it is the source of the flaps*. Given the largely diverse causes of route flaps, we do not claim LRFD can handle all the persistent route flaps well. Instead, we are more interested in illustrating the possibility of employing light-weight algorithms to handle persistent route flaps.

At a high level, LRFD works as follows. It monitors the *local* network events. Both the window-based counting scheme and the penalty-based algorithm can be used to determine if a network event is a persistent one. If so, all the routes affected by the event are excluded from the BGP routing decision process. We will not elaborate on the suppressing mechanism any further. Below we focus on the monitoring mechanism and assume that the window-based counting scheme is employed for the suppression purpose. (See Section 4 for the discussion on the window-based counting scheme.)

6.1 Handling Physical Network Events

Each node i maintains a counter η_j for each of its BGP neighbors j (either an inter-domain BGP neighbor or an intra-domain one). When the corresponding BGP session is (re-)established, η_i is advanced by one.

Now let's see how LRFD handles physical network events that affect the reachability of *local network prefixes*. Regardless of how a node learns a local network prefix d (either by manual configuration or from intra-domain routing protocols), it maintains a counter η^d for the prefix. Whenever the reachability to the network prefix is changed (from being reachable to being unreachable), the node advances the counter by one. (Different from handling other network events, an AS may prefer not to suppress the announcement of the prefix, but rather damp the propagation of the related network events).

6.2 Handling Policy Network Events

For policy-related network events, we focus on disputed BGP routing policies. Varadhan, Govindan and Estrin demonstrated that disputed routing policies could result in persistent route oscillations [16]. To tackle this problem, several pieces of work proposed rules to define safe routing policies to ensure the convergence and stability of Internet routing [4, 6]. Despite of the existence of these kinds of guidelines on defining safe BGP routing policies, we cannot exclude the possibility that disputed routing policies could still exist on the Internet because of, say, misconfigurations. (Note, however, that disputed routing policies have not been observed on the Internet). To handle disputed routing policies, we make the following assumption about BGP behavior: when a node i is able to reach a destination, it will not send an explicit BGP withdrawal message to a neighbor j unless the neighbor is part of the newly selected best route and node i has previously announced a route to the neighbor. This assumption holds in the current BGP protocol [15]. In a stable system, we should not see the persistent occurrence of the above mentioned behavior at node i, i.e., an explicit withdrawal being sent by node i following the receipt of a new route to the destination, which is the signature of disputed routing policies and forms the basis for LRFD to handle policy-related network events.

For simplicity, the following discussions are made with respect to a network destination d. In LRFD, each node i maintains a counter η_j^d for each neighbor j. Node i maintains the routes of *interest* from the neighbor in a data structure R_j^d. R_j^d is updated as follows: Assume now node i receives a route r from a neighbor i. *If receiving this route triggers explicit withdrawal(s) to be sent to some neighbor(s), then* $r \rightarrow R_i^d$. (Note that in RFD+, all the routes are inserted into R_j^d.) If r is already in R_i^d, a route flap is detected and η_j^d is advanced by one.

7 Related Work

The potential adverse side effect of the BGP Route Flap Damping algorithm on the Internet routing convergence time has been speculated in [5, 11]. The work by Mao *et al.* [10] is perhaps the first demonstrating that BGP RFD can indeed exacerbate the Internet routing convergence time. A simple enhancement to RFD was proposed in their work. However, as we discussed earlier, the proposed enhancement may fail in certain cases. As a result, it may still suppress a relatively stable route for a potentially long period of time.

Orthogonal to the work to design better route flap damping algorithms, there have been several recent efforts trying to eliminate or alleviate the BGP path exploration problem [1, 13]. In [13], the authors defined certain AS path consistency rules for identifying infeasible routes, therefore reducing the chances that outdated information will be selected and further propagated in the Internet. Currently, however, at a BGP speaker the consistency rules are only defined for and applicable to the routes learned from the neighbors where one of the neighbors uses another neighbor as the next hop (or downstream node more generally) to a destination. Therefore, it would not be able to eliminate all the BGP path explorations. Note also that certain relationships among the BGP speaker of interest, its neighbors, and the destination network need to be satisfied so that one of the neighbors can use another neighbor as the next hop to the destination. More investigations are needed to determine how common such relationships are in the current policy practices on the Internet, and consequently the effectiveness of the consistency rules. In [1], a conceptually simple enhancement to the current BGP protocol was made to try to flush out the outdated routing information by speeding up the propagation of *bad news* through *undelayed* BGP withdrawal messages. However, this scheme also can not eliminate the propagation of outdated or invalid routing information, therefore there will still be BGP path exploration following a network event. First of all, if a *MinRouteAdver* has been passed when a withdrawal is received, a new route will be selected and announced to the neighbors. However, this newly selected route can be an invalid route. Second, the holding time for a BGP speaker to announce a new route is *MinRouteAdver* (currently 30 secs). If the outdated routing information has not been flushed out (of the Internet) within this time interval, an invalid route can be announced to the neighbors. Moreover, more withdrawal messages can be potentially generated in this scheme compared to the current BGP protocol, which may have an adverse effect on the BGP Route Flap Damping algorithm. Note also that both pieces of work have no effect on BGP recovery path explorations.

Despite this, it is clear that such efforts reduce the degree of BGP path explorations, which simplifies the task of damping flapping routes. However, we believe that our work is still valuable in the sense that it sheds light on the characteristics of BGP path explorations and provides us with more understanding about the problem related to BGP path exploration.

8 Conclusion and Future Work

In this paper we studied the properties of BGP path exploration and what distinguishes it from actual route flaps. We developed a characteristic "signature" of a route flap that could be checked against received updates. Based on this, we developed a new BGP route flap damping algorithm, RFD+, which correctly identifies route flaps while ignoring updates corresponding to path exploration. Thus, relatively stable routes will not be suppressed, which was a problem in the original RFD algorithm as well as in SRFD. We also briefly discussed a simpler, localized route flap damping scheme.

In the future we plan to analyze the BGP updates collected at different BGP route servers in order to understand the prevalence of route flaps and also to evaluate the performance of RFD+ in the real Internet.

Table 4: Advertisements interleaved with withdrawals

Stage	Routing tables	New messages
0	1(*0d, 30d) 3(*0d, 10d, 40d) 4(*0d, 20d, 30d) 2(*0d, 40d)	- (steady state)
	edge (0,d) is down	0→{1, 2, 3, 4}W
1	1(-, *30d) 3(-, *10d, 40d) 4(-, *20d, 30d) 2(-, *40d)	**1→{x, 3}[130d]**, 3→{1, 4}[310d], 4→{2, 3}[420d], 2→{4}W
2	1(-, -) 3(-, -, *420d) 4(-,-,*310d) 2(-,-)	**1→{x}W**, 3→{1}[3420d], 3→{4}W, 4→{3}W
3	1(-, *3420d) 3(-,-,-) 4(-,-,-) 2(-,-)	**1→{x}[13420d]**, 3→{1}W
4	1(-,-,-) 3(-,-,-) 4(-,-,-) 2(-,-)	**1→{x}W**

A Advertisements Interleaved with Withdrawals

In this example we show that a single link failure can result in multiple route advertisements with the same preference interleaved with route withdrawals.

Figure 8 presents the AS-level network topology we will use. We adopt the same conventions as in the example given in 2.3. For simplicity, we assume that the routes announced from the same neighbor have the same preference. Table 4 presents the the BGP routing information updates sent from node 1 to node x after the edge between node d and 0 is down.

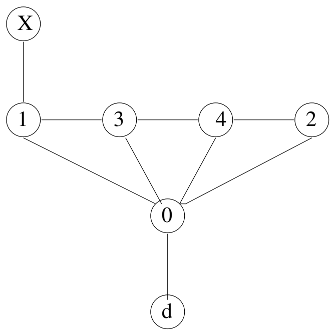

Figure 8: Advertisements interleaved with withdrawals.

From the table, we see that four updates are sent from node 1 to node x regarding the reachability to node d at steps 1 to 4 (they are recorded with bold fonts in the table). Note that the two route advertisements at stages 1 and 3 are interleaved with a route withdrawal.

References

[1] A. Bremler-Barr, Y. Afek, and S. Schwarz. Improved bgp convergence via ghost flushing. In *Proc. IEEE INFOCOM*, San Francisco, CA, April 2003.

[2] R. Bush, T. Griffin, and Z. Mao. Route flap damping: Harmful? In *NANOG*, October 2002.

[3] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless inter-domain routing (CIDR): an address assignment and aggregation strategy. RFC 1519, September 1993.

[4] Lixin Gao and Jennifer Rexford. Stable internet routing without global coordination. In *Measurement and Modeling of Computer Systems*, pages 307–317, 2000.

[5] Timothy Griffin and Brian Premore. An experimental analysis of BGP convergence time. In *Proceedings of IEEE International Conference on Network Protocols (ICNP)*, 2001.

[6] Timothy Griffin, F. Bruce Shepherd, and Gordon T. Wilfong. Policy disputes in path-vector protocols. In *ICNP*, pages 21–30, 1999.

[7] Geoff Huston. Analyzing the internet BGP routing table. *The Internet Protocol Journal*, March 2001.

[8] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed internet routing convergence. In *SIGCOMM*, pages 175–187, 2000.

[9] C. Labovitz, G. Malan, and F. Jahanian. Internet routing instability. *IEEE/ACM Transactions on Networking*, 6(5):515–528, 1998.

[10] Z. Mao, R. Govindan, G. Varghese, and R. Katz. Route flap damping exacerbates Internet routing convergence. In *Proc. ACM SIGCOMM*, Pittsburgh, PA, August 2002.

[11] C. Panigl, J. Schmitz, P. Smith, and C. Vistoli. RIPE Routing-WG recommendations for coordinated route-flap damping parameters, October 2001. Document ID: ripe-229.

[12] V. Paxson. End-to-end routing behavior in the Internet. In *Proc. ACM SIGCOMM*, Stanford, CA, August 1996.

[13] D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang. Improving bgp convergence through consistency assertions. In *INFOCOM 2002*, New York, NY, Jun 2002.

[14] BGP Beacon Info @psg.com. http://www.psg.com/ zmao/bgpbeacon.html.

[15] Y. Rekhter and T. Li. A border gateway protocol 4 (BGP-4). RFC 1771, March 1995.

[16] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks (Amsterdam, Netherlands: 1999)*, 32(1):1–16, 2000.

[17] C. Villamizar, R. Chandra, and R. Govindan. BGP route flap damping. RFC 2439, November 1998.