Improving Network Agility with Seamless BGP Reconfigurations

Stefano Vissicchio, Laurent Vanbever, Cristel Pelsser, Luca Cittadini, Pierre Francois, Olivier Bonaventure

***Abstract*—The network infrastructure of Internet Service Providers (ISPs) undergoes constant evolution. Whenever new requirements arise (e.g., the deployment of a new Point of Presence, or a change in the business relationship with a neighboring ISP), operators need to change the configuration of the network. Due to the complexity of BGP and to the lack of methodologies and tools, maintaining service availability during reconfigurations that involve BGP is a challenge for operators.**

In this paper, we show that the current best practices to reconfigure BGP do not provide guarantees with respect to traffic disruptions. Then, we study the problem of finding an operational ordering of BGP reconfiguration steps which guarantees no packet loss. Unfortunately, finding such an operational ordering, when it exists, is computationally hard. To enable lossless reconfigurations, we propose a framework that extends current features of carrier-grade routers to run two BGP control planes in parallel. We present a prototype implementation and we show the effectiveness of our framework through a case study.

***Index Terms*—Border Gateway Protocol (BGP), configuration, reconfiguration, migration, Virtual Routing and Forwarding (VRF), Ships in the Night.**

I. Introduction

The Border Gateway Protocol (BGP) [1] operates in two different modes. External BGP (eBGP) controls routing information exchange between different ISPs, while internal BGP (iBGP) distributes interdomain routing information among routers in the same ISP. Both eBGP and iBGP configurations are critical for an ISP, as they typically enforce commercial agreements with other ISPs and traffic engineering policies.

During the life of a network, iBGP and eBGP configurations evolve. The organization of iBGP sessions typically need to be periodically modified, e.g., when new iBGP routers are introduced while older ones are either decommissioned or moved to less traffic intensive areas. Also, iBGP configuration changes can be triggered by changes to the underlying Interior Gateway Protocol (IGP). IGP changes are often performed in ISPs [2], e.g., to optimize the usage of network resources by fine-tuning of IGP weights [3]. Unfortunately, IGP configuration adjustments can affect iBGP routing choices, possibly leading to routing and forwarding inconsistencies [4], [5], as well as undesired side effects on internal and external traffic flows [6]. IGP changes may thus require iBGP configuration changes. Similarly, eBGP configuration need to be changed. A typical use case is the provisioning of a new customer, which requires to establish new eBGP sessions on some border routers. As commercial relationships between ISPs change, operators also need to modify their eBGP routing policies. Prominent examples include the so-called "peering wars" that led to the depeering of large ISPs [7]. As a result, routing policies are changed on a daily basis in some networks [8].

The impact of changes to either iBGP or eBGP configuration is hard to predict. Indeed, local changes on one BGP router can affect routing information viewed by remote routers in a domino effect in which the organization of iBGP sessions and message timings may play a critical role. Unfortunately, network administrators lack methodologies and tools to perform reconfigurations with minimal impact on the traffic. Only a few best practices are available (e.g.,[9], [10], [11]), but they typically focus on simple reconfiguration cases. Even worse, current best practices barely take into account the possibility of creating routing and forwarding anomalies during the reconfiguration process.

In this paper, we address the problem of changing the BGP configuration of an ISP with no impact on data plane traffic. We consider both eBGP and iBGP configuration changes. The contribution of this paper is threefold. First, we show that long-lasting routing and forwarding anomalies can and do occur during BGP reconfigurations even when the initial and the final BGP configurations are anomaly-free. We simulated BGP reconfigurations in a Tier-1 network observing that a significant number of anomalies persists for large parts of the reconfiguration process. Such a study exposes the fragility of correct BGP configurations, as different kinds of anomalies can be triggered even by simple changes on a single BGP session. Second, we consider the problem of finding an ordering of configuration changes which guarantees an anomaly-free reconfiguration process. We show that this problem is computationally intractable. Even worse, we present simple cases in which an anomaly-free reconfiguration ordering does not exist at all. Third, we propose a generic framework that enables lossless BGP reconfigurations. Our solution is based on enabling routers to support two independent and isolated control planes, by slightly extending current technology. Our proposal provably prevents both long-lasting and transient problems due to configuration changes. We describe a possible implementation of our framework, and we present a working

Stefano Vissicchio, Laurent Vanbever and Olivier Bonaventure are with ICTEAM, at Université catholique de Louvain, Louvain-la-Neuve, Belgium (e-mail: firstname.lastname@uclouvain.be)

Cristel Pelsser is with Internet Initiative Japan (IIJ), Innovation Institute, Tokyo, Japan (email: cristel@iij.ad.jp)

Luca Cittadini is with Computer Science and Automation Department, Roma Tre University, Rome, Italy (email:ratm@dia.uniroma3.it).

Pierre Francois is with IMDEA Institute, Madrid, Spain (email: pierre.francois@imdea.org)

We thank Randy Bush, Bruno Quoitin, Virginie van den Schriek for their help in improving the paper. This work was partially supported by Alcatel-Lucent. Laurent Vanbever was supported by a FRIA scholarship. Stefano Vissicchio was partially supported by GARR Consortium and made most of the work when being with Computer Science and Automation Department, Roma Tre University.

Step	Criterion
1	Prefer routes with higher `local-preference`
2	Prefer routes with lower `as-path` length
3	Prefer routes with lower `origin`
4	Among the routes received from the same AS neighbor, prefer those having lower `MED`
5	Prefer routes learned via eBGP
6	Prefer routes with lower IGP metric
7	Prefer routes whose BGP next-hop has the lowest `router-id`
8	Prefer routes with shorter iBGP path
9	Prefer the route coming from the neighbor with lower IP address

TABLE I
BGP DECISION PROCESS.

prototype. Finally, we show the effectiveness of our approach through a use case and we study its scalability.

Observe that, beyond addressing current needs of network operators, our framework can be leveraged to achieve additional agility and flexibility, possibly leading to to competitive advantages for ISPs. For example, the ability to frequently change eBGP configuration enables ISPs to adapt routing policies to observed traffic trends and turn off network devices during idle time (e.g., during the night). By rapidly and safely switching preference of routes received from their eBGP neighbors, ISPs can also reduce their transit costs, and take full advantage of services (e.g., Equinix Direct [12]) aiming at more flexible establishment of upstream connectivity.

The rest of the paper is organized as follows. Section II provides some background. Section III states the BGP reconfiguration problem and highlights deficiencies of current best practices. Section IV presents examples in which anomaly-free reconfiguration orderings do not exist. Section V and Section VI describe our framework and its evaluation. Section VII presents related work. Section VIII contains conclusions.

II. BGP AND CONFIGURATION CORRECTNESS

For each destination IP prefix, each BGP router selects its best route among the routes it has received from its neighbors. A route can be seen as a path on the BGP network graph associated with a set of attributes. Since BGP has an internal loop-detection mechanism [1], each route is a node-simple path. Route attributes are used by the BGP decision process [1] to select the best route. The BGP decision process (summarized in Table I) applies steps sequentially until there is only one route left. We refer reader to [11] for a detailed description of the BGP decision process.

The main peculiarity of the BGP decision process is the support for routing policies. BGP policies result from route filtering and route ranking settings. The former consists in defining what routes have to be filtered out, e.g., to prevent the ISP from providing transit towards some destination prefixes. The latter typically involves changing BGP attributes that are relevant for the decision process, especially `local-preference`. The attributes involved in Steps 1 through 4 are globally significant since they are explicitly carried in eBGP routing updates. Conversely, Steps 5 through 9 involve metrics that have local significance for each BGP router. In this paper, we restrict to the typical BGP configuration in which BGP policies are applied at border routers, while iBGP is simply used as a means for route dissemination. In this case, all the routers in the same ISP will eventually select routes that are equally preferred according to Steps 1-4. Hence, unless otherwise specified, we always refer to the subset of routes that are equally preferred according to the first four decision steps. Note, however, that a policy change at one border router can modify this set of routes.

In the following, we call *IGP topology* the weighted graph representing IGP routers and the adjacencies between them. Similarly, we refer to the organization of iBGP sessions as the *iBGP topology*. The original BGP specifications [1] did not allow an iBGP router to relay information to an iBGP neighbor, hence mandating the iBGP topology to be a full-mesh of iBGP sessions. Later, route reflection [13] was introduced to scale the number of iBGP sessions in large networks. With route reflection, the iBGP neighbors of each router are split into three sets: *clients*, *peers* and *route-reflectors*. Each iBGP router propagates its best route according to the following rules: routes learned from a peer or from a route-reflector are relayed only to clients, whereas all other routes are reflected to all iBGP neighbors. In a fully-meshed iBGP network, all iBGP routers are peers. On the contrary, organizing routers in a hierarchy of clients and route-reflectors reduces the number of iBGP sessions, as route-reflectors relay routing information to their clients. In this paper, we assume that each iBGP topology B complies with current design best practices: (*i*) B is a hierarchy where each router can be univocally assigned to a layer; (*ii*) routers in the *top layer* $T(B)$ have no route-reflectors and are all peers; and (*iii*) each router $r \notin T(B)$ has at least one route-reflector. Violating these assumptions results in routing and dissemination issues [4], [14].

We define a BGP configuration C as a tuple (B, I, Υ) consisting of an iBGP topology B, an IGP topology I, and a function Υ that maps each destination prefix to a set of border routers receiving an eBGP route to this prefix. By definition, Υ encodes the eBGP policies. Each element of a BGP configuration influences routing decisions taken by routers. Indeed, the iBGP topology regulates which routes are known by each router, the IGP topology affects route preference at different routers, and Υ determines what routes are available towards each prefix. In the following, we refer to border routers receiving an eBGP route to a prefix as *egress points* for that prefix.

Previous work has shown that both eBGP and iBGP configurations can result in incorrect routing and forwarding, as a consequence of conflicting routing policies. BGP correctness issues can be classified in signaling, forwarding, and dissemination anomalies.

Signaling anomalies [4], [15], [16] or *routing oscillations* occur when BGP routers are unable to converge to a single stable routing state. Oscillations can delay BGP convergence for a possibly indefinite amount of time, wasting resources and negatively impacting traffic. In iBGP, routing oscillations are due to the interaction with the underlying IGP, and can be further classified into two categories: those induced by partial lack of visibility due to the route reflection topology and those induced by the peculiar semantics of the `MED` attribute. We disregard problems due to `MED` specific setting in this paper,

because of space limitations. We show in the following that BGP reconfigurations can be responsible for routing issues even when MED is ignored and BGP policies are very simple. Furthermore, our solution also prevents MED-induced issues during the reconfiguration (see Section V).

Forwarding anomalies [4], [17] occur when routers make inconsistent forwarding choices. Besides inducing suboptimal forwarding and complicating network management and troubleshooting, forwarding anomalies can also disrupt traffic by causing packet deflections and forwarding loops.

Dissemination anomalies [5] or *Loss of prefix Visibility* (*LoV*) consist in improper route propagation that results in some routers having no route to a given prefix. When this happens, packets are either dropped because no route is known or forwarded according to a less-specific route. The former case creates a traffic blackhole, the latter results in inconsistencies between routing and forwarding plane.

We say that a BGP configuration is *anomaly-free* if no signaling, forwarding and dissemination anomalies occur for any destination prefix. In the worst case, each prefix is learned from a different subset of border routers. In this case, we are consistent with previous work on configuration correctness [4], [5].

III. Seamless BGP Reconfigurations

In this section, we define the BGP seamless reconfiguration problem. By analyzing historical configuration changes deployed in a Tier-1 ISP, we show that the problem has practical relevance. Moreover, we show that incremental approaches and current best practices [9], [10] incur the risk of introducing reconfiguration-induced anomalies. Finally, by means of simulations, we quantify the disruptions generated by existing approaches in simple reconfiguration scenarios.

A. Problem Statement

We define a *reconfiguration*, or *migration*, as a sequence of configuration changes that turn an initial BGP configuration into a final one. We will use indexes to denote intermediate configurations. For example, C_t is the BGP configuration at time t. We define two special indexes i and f that refer to the initial and the final time in the reconfiguration, respectively.

Throughout the paper, we assume C_i and C_f to be given as input and to be anomaly-free. Also, the IGP configuration and the eBGP routes to each destination are supposed not to change during the reconfiguration. As a consequence, the combination of egress points for any destination (i.e., Υ) changes only as an effect of local eBGP configuration changes. We show that BGP reconfigurations are hard even when these assumptions hold.

To improve network agility and quickly react to routing changes, we aim at enabling BGP reconfigurations of production networks at anytime, potentially even during peak hours. Performing reconfigurations during maintenance windows would be extremely slow, as maintenance windows are typically short and rare (e.g., few hours per month) because of stringent Service Level Agreements (SLA). To respect such SLAs, simply shutting down and restarting the network with the new configuration is also not viable. In addition, simultaneously overwriting configuration files on all the routers is unpractical, as it is likely to generate huge control plane churn, which, in turn, can overload routers. Moreover, the latter approach does not allow operators to keep the reconfiguration process under control, turning misconfigurations or human errors (e.g., typos) into a management nightmare.

Hence, an *incremental* approach is needed. In this paper, we disregard migrations where router configurations are modified on a per-prefix basis. Indeed, given the size of current BGP RIBs, per-prefix migrations incur severe penalties in the speed and the ease of management of the migration. Hence, we consider reconfigurations in which the final configuration is installed at one router at each step.

We define a migration as *seamless* if for any migration step j, with $i \leq j \leq f$

- C_j is anomaly-free;
- C_j is not subject to unintended traffic shifts.

An *unintended traffic shift* is a change in the best path selected by a router to a given prefix in which the egress point is neither the initial nor the final one. We also talk about an unintended traffic shift when a router switches between the initial and the final egress points *multiple* times. By definition, unintended traffic shifts are peculiar to the reconfiguration problem, that is the reason why they have not been studied in prior work. We consider avoiding unintended traffic shifts as a primary requirement for seamless BGP migrations, since BGP next-hop changes can disrupt traffic engineering policies (e.g., forcing traffic to exit from other continents), adversely impact costs (e.g., swelling traffic flows on transoceanic links), and significantly increase the likelihood of congesting some links (e.g., under-provisioned backup links). Personal communications with operators confirmed that avoiding unintended traffic shifts is among their most relevant concerns.

During migrations that are not seamless, routing and forwarding anomalies occur in intermediate configurations. These anomalies persist until another intermediate configuration (or the final one) is reached, which might require several migration steps. We refer to such persistent anomalies as *migration anomalies*. Migration anomalies can cause disruptive effects, among which forwarding deflections and loops, unintended traffic shifts, traffic blackholes, congestions, unnecessary iBGP churn, and unnecessary eBGP updates which increase the risk of route dampening [18]. On the contrary, we do not consider short-lived protocol-dependent issues, like those occurring transiently during protocol convergence, as they are unrelated to BGP reconfigurations. Nevertheless, our proposed solution also prevents this kind of issues to occur during the reconfiguration (see Section V).

B. Frequency of BGP Reconfigurations

To illustrate the frequency of BGP configuration changes, we analyzed the BGP configurations of approximately 20% of the routers of a Tier-1 ISP, from April 2010 to July 2011. The considered routers were new generation routers progressively added to the network during the considered timeframe. Among those routers, some have been replaced

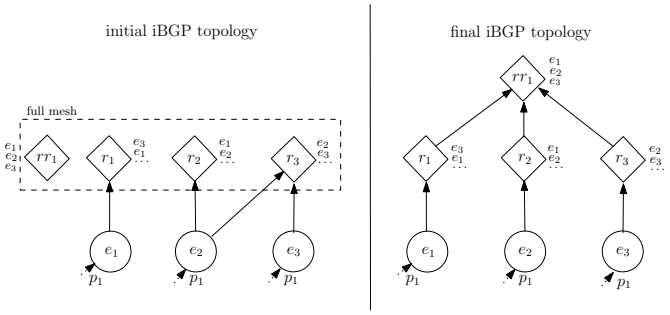

Fig. 1. An example in which the bottom-up strategy, suggested by the current best practices, creates routing oscillations during the reconfiguration.

after their introduction by other routers of a different brand: this happened 17 times. Among the configuration changes, sessions additions and removals were the most common. Sessions additions happened $5,828$ times, encompassing 976 eBGP sessions and $4,852$ iBGP sessions. Session removals were less frequent but still not rare, as they happened 236 times for eBGP sessions and $1,440$ times for iBGP sessions. At each router, eBGP sessions were typically added in groups, while iBGP sessions were mostly added in pairs of redundant sessions with two route-reflectors. By only looking at route-map names, we also registered 41 changes of inbound eBGP policy and 77 modifications of outbound eBGP policy.

Finally, we collected less frequent miscellaneous changes, encompassing the promotion of a router to the role of route-reflector (11 times), AS number modification on an eBGP peer (8 times), and address family enabling (3 times) and disabling (5 times) on eBGP sessions. These numbers testify that reconfigurations of already established BGP sessions are also performed by operators, even if less frequently than the addition or the removal of BGP sessions.

C. Current Best Practices Provide No Guarantees

Currently, network operators can only rely on a few rules of thumb that only apply to simple topological changes, like the replacement of a fully-meshed iBGP topology with a two-layer route reflection hierarchy [9], [10]. In the following, we show that current best practices provide no guarantees on the absence of migration anomalies. To be as general as possible, we consider as current best practice an extension of the procedures proposed in [9], [10] devised after discussions with operators. Such an extension consists in reconfiguring routers, one at the time, on a per-layer basis, in a bottom-up fashion (i.e., starting from the bottom layer up to the top one). Each router r is reconfigured by activating all the sessions r has in the final configuration before shutting down all the sessions r maintains exclusively in the initial configuration.

An example of migration oscillation created by the bottom-up approach is reported in Fig. 1. The graphical convention we adopt in the figure is the same we use for iBGP topologies throughout the paper. Circles represent iBGP routers having no clients, while diamonds represent route-reflectors. Sessions between clients and route-reflectors are drawn as lines terminating with an arrow on the side of the route-reflector. On the contrary, sessions between iBGP peers are represented by double-arrow lines. Short dashed arrows entering a router r and labeled with a prefix p represent the fact that r is an egress point for destination p. Aside each router, we specify route preferences as a list of egress point preferences, in which each egress point represents all the routes propagated by that egress point. Egress point preferences are consistent with the IGP topology. Also, some egress points can be omitted in the list aside any router u if routes from those egress points are guaranteed not to be selected by u. In particular, we omit egress points from u's list if a more preferred egress point e exists from which u is guaranteed to receive a path, e.g., e being a direct client or a direct route-reflector of u.

In the example of Fig. 1, a cyclic preference of routes (i.e., a dispute wheel [15]) exists among r_1, r_2, and r_3. Such preferences can lead to routing oscillations [4]. However, in the initial configuration, the session (e_2, r_3) ensures that r_3 always receives a route from its most preferred egress point, forcing a stable state to be eventually reached. The final configuration is also oscillation-free since rr_1 breaks the cycle of route preferences by steadily selecting the route from e_1. A seamless migration can be achieved by reconfiguring r_3 in the first step. Indeed, after its reconfiguration, r_3 will not learn the route propagated by r_2 throughout the rest of the migration, since r_3 will have sessions only with e_3 and rr_1, which are guaranteed to always select the route from e_3 and e_1 respectively. This breaks the cycle of route preferences, ensuring no migration oscillation. On the contrary, the bottom-up approach does create migration oscillations since it requires to reconfigure e_1, e_2, and e_3 before all the other routers. After the reconfiguration of e_2, (e_2, r_3) is removed, immediately raising routing oscillations [4] due to the cyclic preference of routes among r_1, r_2, and r_3, The problem is fixed only when migrating middle layer routers.

Similar examples hold for other migration anomalies, like forwarding loops and unintended traffic shifts [19].

D. Quantitative Analysis

To estimate the issues generated by simple migration approaches on realistic topologies, we simulated BGP reconfigurations on a Tier-1 network consisting of roughly 100 iBGP routers organized in three layers of route reflection. We performed two kinds of experiments: iBGP topology changes and eBGP policy modifications.

The first kind of experiments consisted in replacing an iBGP full-mesh with the given route reflection hierarchy. We evaluated completely random router migration orderings (e.g., the one generated by a script which naively iterates on all the routers), random orderings in which top layer routers are reconfigured at the end, and bottom-up orderings as prescribed by current best practices. We denote these strategies as *RND* (Random), *RBT* (Random But Top), and *BTU* (Bottom-Up), respectively. For each strategy, we run 50 different experiments, each corresponding to a different ordering. For each experiment, we used C-BGP [20] to compute all the BGP routing tables in the intermediate configurations.

Fig. 2 plots the fraction of migration steps during which different types of anomalies occurred. A point (x, y) in the

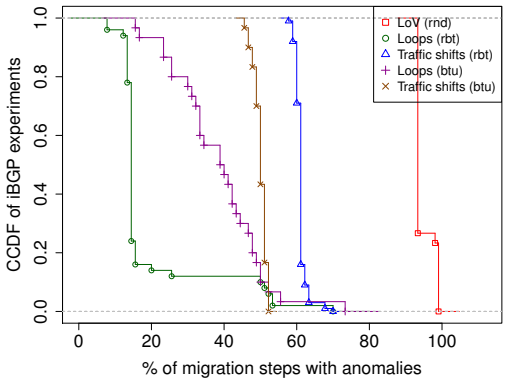

Fig. 2. Percentage of the migration process affected by anomalies during a full-mesh to route reflection reconfiguration on a Tier-1 topology.

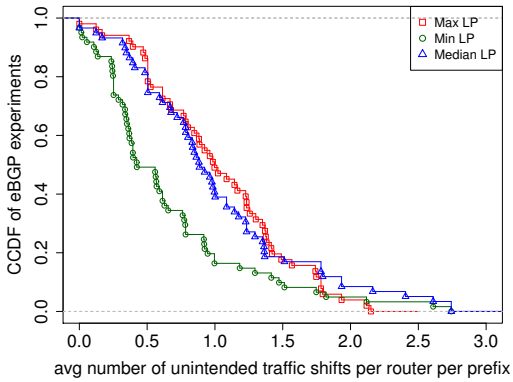

Fig. 3. Average number of unintended traffic shifts that each router experienced per-prefix during eBGP policy changes in a Tier-1 network.

graph means that $(100 * y)\%$ of the orderings of a given strategy exhibited a given anomaly for at least $x\%$ of the migration steps. *RND* orderings almost always triggered Loss of prefix Visibility (LoV) at some iBGP router, for some prefixes. This makes random orderings clearly not viable in practice. *RBT* migrations were not subject to LoVs, however they were responsible for several migration issues. Indeed, in more than 90% of the experiments, loops occurred during more than the 10% of the migration steps. Even worse, unintended traffic shifts occurred during more than 55% of the *RBT* migration process in almost all the experiments. *BTU* performed slightly better than *RBT* on traffic shifts, but, surprisingly, *BTU* creates more forwarding loops than *RBT* on average on the considered network. Indeed, in almost 60% of the experiments, loops were raised during more than 35% of the migration steps. We stress that each traffic shift can affect several prefixes, among which the prefixes that drive the vast majority of traffic [21]. Hence, most of the user traffic may be moved (possibly multiple times) between several egress points, causing relevant service degradations. Our experiments also show that performance degradation can be long-lasting. As a rule of thumb, assuming that a migration step takes about 3 minutes (e.g., for ensuring BGP convergence), then having a loop for 35% of the migration steps translates to losing traffic (towards some prefixes) for about 100 minutes.

In the second kind of experiments, we measured the amount of unintended traffic shifts created by changes of eBGP policies. In each of those experiments, we modified the value of the `local-preference` assigned to the routes received by a given eBGP neighbor. Our data set consisted of the C-BGP [20] model of the Tier-1 along with a dump of the Adj-RIB-In from some top-layer route reflectors. In order to focus on significant traffic shifts, we restricted our analysis to the 940 prefixes that together were responsible for 80% of the traffic [22]. We then identified the 50 eBGP neighbors announcing at least one of that 940 prefixes and having more than one eBGP peering with the Tier-1. We finally computed the unintended traffic shifts caused by different reconfiguration orderings by comparing routing tables at each step with the initial and the final ones. In each experiment, we considered one among the 50 eBGP neighbors previously identified and we changed the `local-preference` value to different final values, which we denote as LP_f. Namely, in different experiments, we set LP_f to the minimum, maximum, and an intermediate (median) value among those found in the initial configuration. These scenarios correspond to turning a neighboring ISP into a provider, customer and peer, respectively. For simplicity, we assumed the Tier-1 to initially apply the same `local-preference` on all the eBGP peerings it keeps with the same neighbor. For each of experiment, we repeated traffic shift measurements for 5 different orderings in which border routers are reconfigured.

Fig. 3 shows the complementary cumulative distribution of the average number of unintended traffic shifts per router. Each point in the plot corresponds to an experiment involving a different neighboring ISP, a different value of LP_f, and a different ordering. When LP_f is set to the median or maximum value, on average, 50% (20%, resp.) of the routers experience at least 1 (1.5, resp.) unintended traffic shift for each prefix announced by the ISP considered in the experiment. In some experiments, we recorded more than 2 and 2.5 unintended traffic shifts on average per router per prefix, respectively. We expect these results to be a source of concerns for network operators, especially if they have to change eBGP policies applied to neighbors announcing the few prefixes that drive the vast majority of the Internet traffic [21]. Lowering the `local-preference` to the minimum value creates less traffic shifts on average than setting LP_f to the maximum or the median value. In fact, contrary to the other two scenarios, routes affected by setting LP_f to the minimum value never attract additional traffic, and can only be de-selected by routers that preferred them before. Still, in few experiments, the average number of unintended traffic shifts is more than 2.5 per router per prefix.

IV. An Algorithmic Approach is not Viable

Given that reconfigurations are frequent and that they can have a significant impact on traffic (see Section III), we now study the problem of finding a migration ordering that ensures seamless reconfigurations.

Unfortunately, this reconfiguration problem is computationally hard. It should be noted that the complexity of the recon-

figuration problem cannot be derived from the computational intractability of assessing the correctness of a single BGP configuration [4], [5]. For instance, the correctness of the initial and final configurations might theoretically be leveraged to formulate easy-to-check sufficient conditions for seamless migrations. Such conditions would enable the design of efficient algorithms that preserve correctness with no need to inspect each intermediate configuration. However, despite our assumption of anomaly-free initial and final configurations, we proved in [19] that finding an operational ordering that guarantees no migration anomalies is $\mathcal{NP}$-hard in both the iBGP and the eBGP cases. Indeed, we showed a polynomial-time reduction from 3-SAT problem, based on mapping boolean assignments of a 3-SAT instance to reconfiguration orderings.

Even worse, in this section we present examples in which every operational ordering leads to migration anomalies. We first tackle iBGP topology changes, then we address the problem of changing eBGP policies.

A. iBGP Topology Changes

The problem of changing the iBGP topology can be formalized as follows. We refer to an operational ordering that guarantees a seamless migration as *seamless ordering*.

Session Ordering Computation Problem (SOCP): given the initial and final iBGP topologies B_i and B_f, compute a seamless ordering in which to add sessions in $B_f \setminus B_i$ and to remove sessions in $B_i \setminus B_f$.

To be as general as possible, we allow multiple sessions involving the same router to be simultaneously added or removed at each migration step. This closely reflects the degree of freedom that operators have. Indeed, multiple sessions involving the same router r can be simultaneously reconfigured by changing the configuration of r. On the contrary, admitting simultaneous changes on arbitrary sessions is less realistic, since perfect synchronism between routers must be assumed for both configuration commits and processing of BGP updates at multiple devices. Moreover, allowing simultaneous operations on different routers overcomplicates controlling the reconfiguration, e.g., if a commit fails.

Observe that SOCP does not take into account possible changes in the interdomain routing. Indeed, given an initial configuration $C_i = (B_i, I, \Upsilon)$, Υ is assumed not to change throughout the migration process. In the following, we show that even if eBGP is stable, there are cases in which a seamless ordering does not exist. Even worse, there are cases in which i) every reconfiguration ordering is not oscillation-free; ii) every reconfiguration ordering is not LoV-free; iii) every reconfiguration ordering is not deflection-free; iv) every reconfiguration ordering is subject to unintended traffic shifts. It is simple to extend those examples to cases in which no reconfiguration ordering is free from different kinds of anomalies, e.g., some orderings creates migration oscillations while others forwarding loops. In the following, we show two examples in which migration oscillations and migration loops cannot be avoided, respectively. Similar examples for the other kinds of migration anomalies can be found in [19].

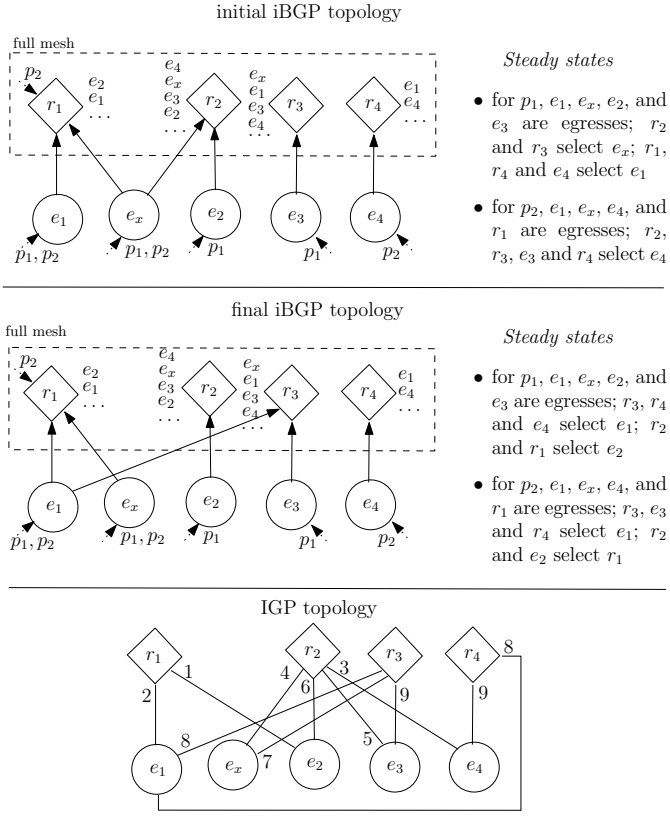

Fig. 4. TWICE-BAD gadget, an iBGP topology change case in which an oscillation-free reconfiguration ordering does not exist.

Fig. 4 depicts an example in which every reconfiguration ordering creates a permanent oscillation in an intermediate configuration. Observe that both the initial and the final configurations are oscillation-free. Indeed, it is easy to check that the configurations are guaranteed to converge to the stable states reported in Fig. 4.

However, an oscillation occurs in every migration ordering. Indeed, since sessions to be added and removed have no routers in common, we have only two cases.

- $add(e_1, r_3)$ before $remove(e_x, r_2)$. Immediately after the addition of (e_1, r_3), r_2, r_3, and r_4 respectively prefer paths $(r_2\ r_4\ e_4)$, $(r_3\ r_2\ e_x)$, and $(r_4\ r_3\ e_1)$ for p_2.
- $remove(e_x, r_2)$ before $add(e_1, r_3)$. Immediately after removal of (e_x, r_2), r_1, r_2, and r_3 respectively prefer paths $(r_1\ r_2\ e_2)$, $(r_2\ r_3\ e_3)$, and $(r_3\ r_1\ e_1)$ for p_1.

In both cases, a cyclic preference of routes prevents iBGP to converge to a stable state [4] for either p_1 or p_2. We experimentally confirmed that no oscillation-free ordering exists, by emulating the initial, the final, and the two possible intermediate BGP topologies in a virtual environment.

Similarly to control plane issues, in some cases forwarding anomalies occur in every reconfiguration ordering, even if B_i and B_f are deflection-free. Consider the example in Fig. 5. In B_i, all routers but s send traffic to e_0, since r_1 does not receive the route announced by e_1, and r_2 prefers routes from e_0 over those from e_1. Similarly, in B_f, all routers but s select the route received from e_1, since r_2 does not receive the route announced by e_0, and r_1 prefers routes from e_1 over those

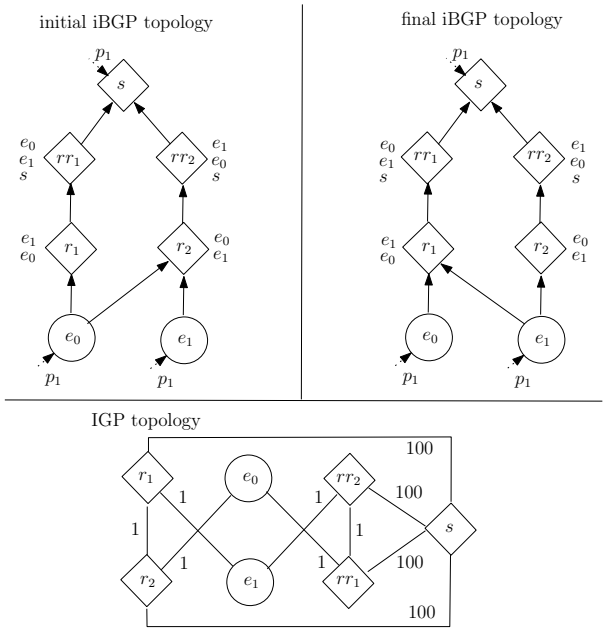

Fig. 5. PYLON gadget, an iBGP topology change case in which a loop-free reconfiguration ordering does not exist.

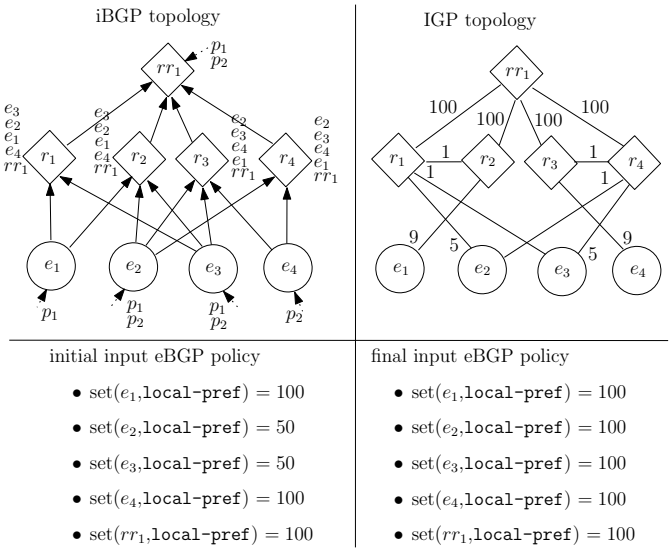

Fig. 6. CAROUSEL gadget, an eBGP policy reconfiguration case in which forwarding loops occur in every reconfiguration ordering.

from e_0. However, one of the following cases apply to the intermediate configuration in every reconfiguration ordering.

- $remove(e_0, r_2)$ before $add(e_1, r_1)$: r_1 and r_2 are forced to select $(r_1\ e_0)$ and $(r_2\ e_1)$ respectively, hence a loop occur between r_1 and r_2 (see the IGP topology).
- $add(e_1, r_1)$ before $remove(e_0, r_2)$: because of path preferences, r_1 and r_2 will select $(r_1\ e_1)$ and $(r_2\ e_0)$ respectively. As a consequence, rr_1 and rr_2 will select $(rr_1\ r_1\ e_1)$ and $(rr_2\ r_2\ e_0)$ respectively, giving rise to a loop between rr_1 and rr_2 (see the IGP topology).

In both cases, a migration loop occur.

B. eBGP Policy Changes

Similarly to the iBGP topology change problem, the eBGP policy change problem is stated as follows.

Policy Ordering Application Problem (POAP): given the initial and the final routing policies, compute an ordering in which to apply the new policies on routers while guaranteeing a seamless migration.

Basically, POAP boils down to studying how intermediate policies affect the set of routes injected in iBGP. Indeed, both the IGP and the iBGP topologies are assumed not to change during the reconfiguration, that is $C_i = (B, I, \Upsilon_i)$ and $C_f = (B, I, \Upsilon_f)$, with possibly $\Upsilon_i \neq \Upsilon_f$. In intermediate configurations, function Υ can also be different from both Υ_i and Υ_f. Hence, our formulation of the problem encompasses both cases in which i) the set of egress points for a given prefix changes only in the intermediate configurations; and ii) the set of egress points for a given prefix changes also between the initial and the final configurations.

Assuming again that eBGP is stable throughout the migration, the Υ function in intermediate configurations depends only on the reconfiguration ordering. Again, migration anomalies cannot be avoided in some cases, even if both the initial and the final configurations are anomaly-free. Fig. 6 shows an example in which migration loops cannot be avoided. Consider prefix p_1. In the initial configuration, e_2 and e_3 do not select eBGP routes, because of the `local-preference` settings, and e_1 and rr_1 are the only two egress points for p_1. Hence, r_1, r_2, e_2, and e_3 select the route from e_1 because of egress point preferences, while r_3, r_4, and e_4 select the route from rr_1 because it is the only route they receive. The IGP topology ensures that no deflection occurs. In the final configuration, all e_i with $i = 1, 2, 3$ and rr_1 are egress points for p_1. Also, r_1 and r_2 select e_3, and r_3, r_4, and e_4 select e_2, because of egress point preferences. Since r_1 and r_2 (r_3 and r_4, respectively) agree on the egress point to use, no deflection occurs. Similar arguments apply to p_2. However, if e_2 is reconfigured before e_3, then r_2 starts receiving and selecting the route from e_2, because of egress point preferences. On the contrary, r_1 keep selecting the route from e_1, as it does not receive the route from e_2. Thus, a forwarding loop is generated between r_1 and r_2. A symmetric loop occurs for p_2 between r_3 and r_4 if e_3 is reconfigured before e_2.

Similarly, all the other kinds of migration anomalies can be created by changing the eBGP configuration, unless the BGP topology is guaranteed to be signaling, forwarding and dissemination correct for any possible set of egress points [4], [5]. Observe, however, that checking for any correctness property to be enforced for any combination of egress points has been shown to be an $\mathcal{N}P$-hard problem [4], [5].

In addition, there are cases in which unnecessary traffic shifts cannot be avoided by any configuration ordering. The example in F.ig. 7. represents a scenario in which preference of eBGP routes R_1 and R_2 has to be lowered, e.g., because a commercial relationship with a neighboring ISP has changed. In this case, an unnecessary traffic shift occurs in every migration ordering. Indeed, in both C_i and C_f, traffic towards

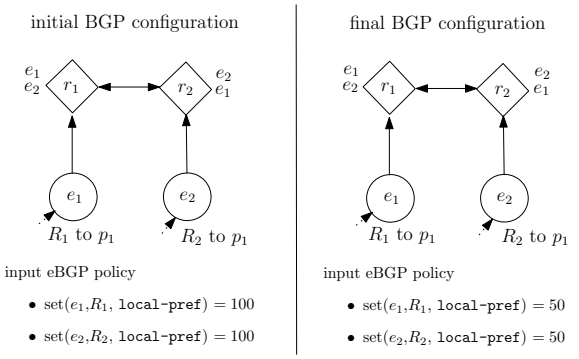

Fig. 7. FRAGILE gadget, an eBGP policy reconfiguration case in which unintended traffic shifts occur in every reconfiguration ordering.

p_1 is load-balanced among e_1 and e_2, since r_1 and e_1 use R_1, while r_2 and e_2 use route R_2. However, if e_1 is migrated first, then all iBGP routers start preferring R_2 because the route is temporarily assigned a higher `local-preference` with respect to R_1. Hence, r_1 and e_1 are subject to an unnecessary traffic shift that holds until routing policy is changed on e_2. A symmetrical traffic shift occurs if e_2 is migrated before e_1.

V. A General Solution for BGP Reconfigurations

Section IV shows that seamless BGP reconfigurations cannot be always achieved by just adding and removing sessions. Intuitively, the problem is that local changes can unpredictably impact routing decisions at remote iBGP routers.

We argue that additional configuration tools are needed to build a general approach enabling seamless migrations. We propose to run two distinct control planes on all iBGP routers, as it is normally suggested for IGP reconfigurations (e.g., [10]). The co-existing control planes run different configurations, i.e., one control plane runs the initial configuration and the other runs the final configuration. To avoid control plane anomalies, the two control planes work in isolation (no route leakage from one plane to the other), and are already converged to a stable state when starting the reconfiguration. To avoid data plane anomalies, our solution specify what control plane must be used network-wide for packet forwarding. We refer to this approach as BGP *Ships-In-The-Night* (SITN).

A. Requirements and Challenges for Two Control Planes

The main advantage of BGP SITN is that it allows us to reconfigure a single router without affecting routing decisions of other routers. Indeed, running the initial and the final configurations in separate control planes enables each router to compute both the initial and the final BGP routing tables (RIBs). Then, a router reconfiguration just mandates the router to forward traffic according to the final RIB instead of the initial one. Unfortunately, current routers cannot natively support multiple BGP routing processes on the same set of eBGP routes.

From an abstract point of view, the following functionalities are needed in order to implement BGP SITN:

- co-existence of multiple isolated routing processes on the same router; and
- independent propagation of all routes to all routing processes within each router.

In order to simulate co-existence of multiple routing processes on the same router, we can leverage the Virtual Routing and Forwarding feature [23] available on commercial devices. This feature is currently used as a basis for MPLS L3VPNs and BGP multi-topology.

Basically, Virtual Routing and Forwarding creates isolated namespaces for prefixes by tagging each set of prefixes with a route distinguisher. Two routes having distinct route distinguishers cannot be compared, and can co-exist in the routing table. By default, namespaces do not share any route, though route import and export mechanisms enable leakage of best routes to given prefixes from one namespace to another. Each network interface of the router can be assigned to a single namespace in such a way that forwarding depends both on the destination prefix and on the ingress interface. In the following, we will refer to each namespace as a VRF.

To run two control planes at the same time, we would use an *initial VRF* with the initial configuration, and a *final VRF* with the final configuration. Unfortunately, because of the one-to-one mapping between interfaces and VRFs, routes learned from external peers are injected in a single VRF. This prevents independent propagation of external routes to all the VRFs, since only the best routes can be leaked from one VRF to another. A workaround to propagate all the external routes to all the VRFs is to configure multiple parallel eBGP peerings. However, this solution is unpractical as it unnecessarily duplicates eBGP peerings and requires coordinated configuration changes on both sides of those peerings.

Forwarding inconsistencies must also be avoided. If two routers disagree about which VRF a packet should be assigned to, the network could experience forwarding deflections, loops and congestion, hence packet loss [2]. Thus, correct forwarding requires that every router on the data path of a packet forwards it according to the same VRF. For this reason, packets need to be *tagged* with VRF information.

We distinguish between explicit and implicit tagging. *Explicit tagging* involves modifying the packet to encode additional information which is processed at every router. Traffic *encapsulation* mechanisms, e.g. MPLS or GRE, are examples of explicit tagging. Conversely, *implicit tagging* requires no change to data packets. Tags are inferred and assigned to packets on the basis of information at lower layers in the protocol stack, e.g., the logical interface which receives the packets. An example of implicit tagging is what is commonly known as VRF-lite. In a VRF-lite based network, routers are configured with multiple logical interfaces on the same links and separate IGP instances are run in each VRF. In this case, the VRF tag is implicitly assigned to each data packet according to the destination MAC address of the frame.

B. Proposed Solution

The BGP SITN approach requires three key components: a dispatching mechanism to propagate all the external routes to multiple namespaces, a front-end interface which propagates iBGP updates from one "active" namespace to the eBGP

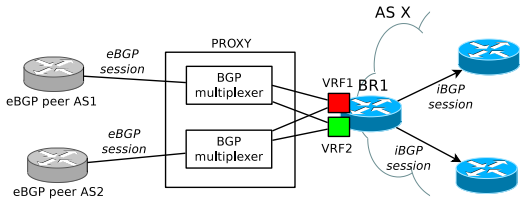

Fig. 8. Architecture of our solution.

neighbor, and a tagging mechanism, either implicit or explicit. While we can leverage multiple tagging mechanisms (MPLS and VRF-lite, for instance), we currently lack support for the other two key components.

To this end, we propose to interpose a *proxy* component between each border router and its eBGP peers, as depicted in Fig. 8. The architecture of the proxy is similar to the one of BGP-Mux [24] in that the proxy maintains an eBGP peering with external neighbors and one iBGP client session per VRF configured on the border router. However, we extend the architecture proposed in [24] to support the concept of "active" namespace and the selective propagation of iBGP updates to the eBGP neighbor. Indeed, the proxy distinguishes one *active VRF* from several *passive VRFs*. All VRFs receive external routes from eBGP peers, but only information in the active VRF is considered when sending eBGP updates to external neighbors. While the proxy can be implemented as a standalone device, we envision its functionality to be built directly inside border router to facilitate reconfigurations.

Since the proxy maintains eBGP peerings on behalf of a border router, it needs to be configured. The proxy configuration is simple as consists in the following information.

- the address of each eBGP peer;
- for each VRF, the name of the VRF and the address of the interface on the border router which is assigned to that VRF; and
- the name of the active VRF.

Finally, to implement the tagging mechanism, the proxy exploits the third-party BGP next-hop feature that implicitly maps packets from external neighbors to the active VRF. More precisely, whenever the active VRF is changed, the proxy advertises to its eBGP peers a change of the BGP next-hop, forcing them to send data packets to the interface bound to the new active VRF. For this reason, the proxy does not need any packet forwarding ability.

The ability of switching a VRF from active to passive makes it easy to deploy changes at border routers, e.g., changing eBGP policies. Reconfigurations that involve iBGP topology changes need extra care. Sessions that are present in both the initial and the final configuration are left untouched. Whenever an iBGP session has to be added to the final configuration or removed from the initial one, we run multiple iBGP sessions in parallel. Whenever the type of an iBGP session has to be changed (e.g., from a client session to a peer session), we run two iBGP sessions in parallel between the same pair of iBGP routers, using two different pairs of loopback interfaces. We prevent routes in the initial (final, resp.) VRF to be propagated over iBGP sessions that are not in the initial (final, resp.) configuration by means of route-maps.

A reconfiguration step in BGP SITN simply consists in switching the active VRF at one proxy. This has two main consequences: first, packets from eBGP neighbors start to be forwarded according to the new active VRF; second, the proxy announces the routes in the new active VRF to its eBGP neighbors. As one proxy per border router exists, the reconfiguration can be performed one border router at a time.

C. Key Benefits

A primary benefit of our solution is that it guarantees seamless BGP migrations, as proved by the following theorem.

Theorem 1: Our solution ensures seamless migrations.

Proof: First, we consider control plane anomalies. BGP SITN ensures that the two control planes run network-wide in isolation, meaning that routes received by each router in each control plane coincide with the routes that the router receives in either the initial or in the final configuration. Also, the selection of the active VRF on each router has no impact on any of the two control planes. Hence, absence of control plane anomalies follows from the assumption that both the initial and the final configurations are correct.

Regarding data plane anomalies, both deflections and unintended traffic shifts are prevented by the tagging mechanism. Indeed, whatever is the active VRF on any proxy, the tagging mechanism ensures that every router in the network will use the same VRF to forward any data packet. Hence, traffic will be forwarded over a forwarding path which is either the path used in the initial configuration or the one followed in the final configuration. The correctness of the initial and final configurations ensures no data plane anomalies. ■

Observe that our approach is also suitable to deal with routing and forwarding issues we have disregarded for the sake of simplicity. Primarily, our approach requires no extra logic to adapt to changes in both the control plane (e.g., eBGP changes) and the data plane changes (e.g., physical failures). Indeed, thanks to the proxy and the isolation principle, both SITN control planes react to routing changes independently. Also, our approach avoids potential issues due to the `MED` attribute, as the initial and the final control planes are isolated and assumed to be anomaly-free. Finally, our approach prevents transient anomalies, like protocol convergence issues, possibly caused in the incremental approach at each reconfiguration step. Indeed, the two control planes running in BGP SITN do not need to converge at each migration step, but only before starting the reconfiguration and possibly after external changes like eBGP or physical changes.

The main drawback of our approach is the additional load imposed to routers, since they have to support two control planes. In Section VI we show that current carrier-grade routers can sustain this additional load. For legacy network devices and to narrow the additional iBGP churn, it is also possible to group prefixes in few sets, and apply the migration procedure for each of these sets. Indeed, the design of the proxy easily enables to set different active VRFs for different groups of prefixes. We evaluate the trade-off between quickness and scalability of our approach in Section VI.

VI. Evaluation

In order to show the feasibility and effectiveness of our solution, we implemented a prototype that can perform seamless reconfigurations. We use our prototype to perform a use case, and we evaluate the scalability of our solution. Finally, we qualitatively compare our approach with alternative proposals.

A. Implementation

The system is based on an extended version of the provisioning system presented in [2] to which we added support for VRFs and route-maps. At each migration step, our system reconfigures one border router by interacting with the corresponding proxy and switching the active VRF on it.

We implemented the proxy as a standalone script of about 400 lines in Perl. Observe that the proxy can be interposed between a border router and an eBGP neighbor without tearing down the BGP peering by taking advantage of the BGP graceful shutdown mechanism [25].

Our prototype proxy has some known limitations: first, it requires the ability to define logical interfaces on the border router; second, it requires the proxy, the external neighbor and the border router to share the same layer 2 infrastructure. However, these limitations could be easily avoided if the proxy were directly integrated in the router operating system. Given the simple architecture of the proxy, we believe such an integration to be possible on commercial routers.

B. Case Study

Based on our prototype implementation, we simulated a full-mesh to route reflection reconfiguration of Geant, the pan-European research network. We run the simulation in a virtual environment on a Sun Fire X2250 (quad-core 3GHz CPUs with 32GB of RAM). Routers were emulated using a major router vendor operating system image.

In our case study, we assumed Geant to offer MPLS L3VPN services, with VRFs (one per customer) configured on the border routers, and MP-BGP running in the core of the network. We built the IGP and the iBGP configurations consistently with the layer 2 topology of Geant [26]. The IGP configuration consists of a single area where link weights are inversely proportional to their speed. The route reflection configuration was designed on the basis of the geographical position of the routers, a design practice commonly used by network operators [11]. The route reflection top layer is composed of four routers, namely, DE, FR, NL, and UK. The routers having a fiber link to one top layer router were assigned to the middle layer. The remaining routers were added to the bottom layer. Each router in the middle and in the bottom layer has two route reflectors belonging to the layer immediately above, mimicking redundancy best practices.

To identify the set of sites at which different customers connect to Geant, we used real-world BGP updates. We found 16 different sets of egress points that receive BGP routes for the same prefix. We mapped those sets on different customers of Geant, and we injected through each of them a different *summary prefix*, representing all the prefixes for the customer.

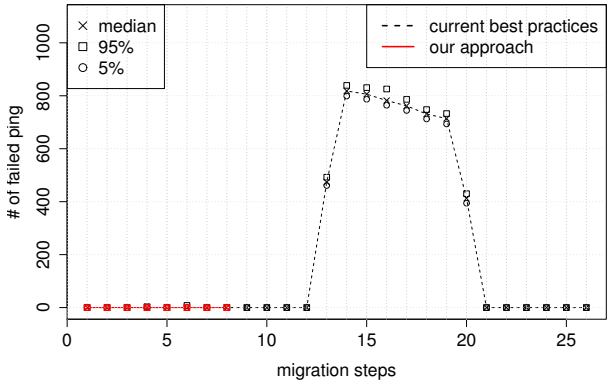

Fig. 9. Using our system, no packet was lost when converting the Geant network from an iBGP full-mesh to a route-reflection hierarchy. On the contrary, significant traffic losses occurred with current best practices.

Then, we evaluated two different reconfiguration strategies. In the first experiment, we reconfigured the network using our system. In particular, we configured the initial and the final VRFs on each border router, and we added final iBGP sessions to the iBGP configuration. Two route-maps per router ensured correct propagation of routes on the initial and final iBGP topologies. Then, we proceeded one border router at a time. To migrate a border router, we activated the final VRF on the proxy. When the final VRF is used on all the border routers, we remove the initial iBGP sessions, the initial VRFs and both route-maps from the routers. In the second experiment, we followed the current best practices [9], [10]. In particular, for each router to be migrated, we first activated the sessions with its route reflectors, then we waited for route propagation, and finally we removed the initial sessions. We applied a bottom-up reconfiguration order. Within each layer, we picked routers according to the alphabetical order of their names. We repeated each experiment 30 times to minimize the impact of factors beyond our control (e.g., related to the virtual environment). To measure possible traffic disruptions, we injected ICMP echo request from each router towards each summary prefix throughout the migration process.

Fig. 9 reports the 5th, the 50th, and 95th percentiles of ICMP packets lost during each migration step. Note that our approach terminates the migration in fewer steps (only 8 instead of 26) with respect to current best practices, since the new configuration is used network-wide after the reconfiguration of the border routers (only a configuration cleaning step is needed for the other routers in the network). This makes the reconfiguration process quicker.

Using our framework, no packet was lost. On the contrary, current best practices induced forwarding loops between reconfiguration steps 13 and 20, hence packets were lost during approximately 30% of the migration. We found that 7 routers lost packets because of loops to two summary prefixes. Together, these two summary prefixes corresponded to more than 60% of all the prefixes known by routers in Geant. Even worse, the fact that discovered loops affected Equal Cost Multi-Paths would have severely complicated possible debugging activities.

All the configurations that we generated, along with the

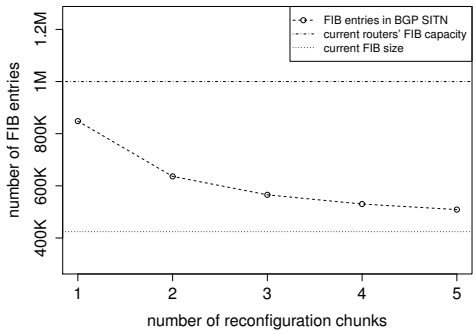

Fig. 10. Scalability evaluation of our solution with respect to FIB size.

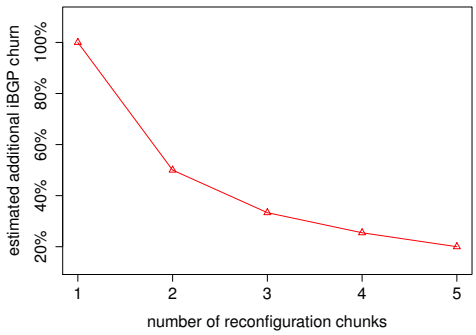

Fig. 11. Scalability evaluation of our solution with respect to churn.

IGP and iBGP topologies, are available online [27].

C. Scalability

We now estimate the overhead of our approach in terms of additional router memory and CPU processing power needed to maintain two control planes. Regarding memory, we focus on the FIB size as RIBs can be easily scaled by adding low cost RAM components. On the other hand, CPU overhead is mostly due to computing the BGP best path twice (once for each control plane), which increases the iBGP churn, i.e., the number of iBGP updates.

Although sharing memory and data structures across multiple VRFs might be a significant performance improvement (e.g., it would compress repeated BGP attributes across VRFs), we find that routers currently store a separate copy of the RIB and the FIB for each VRF. Hence, activating BGP SITN would double the number of FIB entries. This is not a problem for current routers, as shown in Fig. 10. Indeed, the FIB capacity of routers was estimated in at least 1 million FIB entries in 2009 [28] (see the horizontal dashed and dotted line in Fig. 10), while the current FIB size of a typical Internet router is about 425,000 FIB entries as of 28 May 2012 [29]. Moreover, the FIB size can be reduced by dividing the prefixes in n groups and migrating one group at a time, as described in Section V-C. This way, the reconfiguration is divided in n macro-steps or chunks. At each migration step, the total number of FIB entries will be $(1+1/n)$ times the original FIB size, thus reducing the amount of additional memory needed, at the cost of multiplying the number of migration steps by n. The dashed line with circle points in Fig. 10 shows that a good trade-off can be achieved for $n = 2$ or $n = 3$.

Very similar considerations hold for BGP churn: by grouping prefixes in n sets, we can trade speed of the migration process for better scalability. We performed the following analysis. Since both the initial and the final configurations are correct, the number of iBGP updates generated by a single eBGP update must be finite. To be independent of the given iBGP topologies, we assumed that the iBGP churn is proportional to the eBGP churn. Observe that this is a pessimistic assumption as several eBGP updates could generate no iBGP update in one of the two control planes or both. Indeed, no iBGP updates are generated when the received eBGP update does not change the best route selection at the egress point, e.g., in case of duplicate eBGP updates. We collected all the eBGP updates from [29] during May 2012, and we estimated the additional churn introduced by SITN with respect to the churn generated in a single control plane. We used a simple greedy heuristic to divide prefixes in n groups: iteratively, we picked the most churny prefix not yet assigned to any group and we added it to the group having the least total number of BGP updates. Fig. 11 shows the result of such an analysis for a single route collector from [29]. We found very similar results for all the other collectors from [29]. In particular, Fig. 11 plots the additional iBGP churn experienced in SITN with respect to the maximum churn between the initial and the final configurations. Let n be again the number of reconfiguration chunks. If $n = 1$, then the iBGP churn in SITN is the sum of the churn in each control plane. In the worst case, the iBGP churn is equal in the initial and final control planes, and the churn increase due to our solution is 100%. As n increases, however, the red solid line in Fig. 11 shows that the additional BGP churn generated by BGP SITN drops quickly.

The results above suggest that our solution can be deployed in today's networks. In particular, we stress that operators providing MPLS VPN services already have most of the machinery in place to implement BGP SITN. Others should weigh the augmented network agility against the cost of introducing new technologies to configure two control planes. We believe that the long term gain in network agility can motivate operators to bear the initial deployment cost.

D. Comparison with Alternative Solutions

A possible alternative to our solution consists in configuring static routes that match the initial configuration and ensure consistency throughout the migration. Static routes can be eventually removed when the final configuration has been deployed. Unfortunately, this approach has severe drawbacks. First, detaching the control plane from the data plane makes the network unable to react to BGP routing changes (e.g., the withdrawal of a BGP route). Second, our results in Section IV indicate that removing the temporary static routes after BGP has converged can result in forwarding anomalies. Finally, adding a significant number of static routes overcomplicates management and troubleshooting.

In principle, recently proposed techniques to simplify BGP management can be leveraged to facilitate the reconfiguration

process. For example, one might think to rely on platforms that centrally compute BGP routes (e.g., [30]), or mechanisms that separate the control plane from the data plane (e.g.,[31]). On one hand, those techniques would simplify the understanding of intermediate routing states, and help avoiding control plane anomalies. On the other hand, however, they suffer from problems intrinsic to centralized approaches, especially scalability and resiliency of the centralized platform. Also, quickly react to data plane changes (e.g., link and router failures) may be challenging. Finally, deploying the centralized component while avoiding routing and forwarding inconsistencies can be seen as just another seamless reconfiguration problem.

VII. Related Work

Considerable effort has been devoted to BGP configuration correctness [4], [14], [32] and iBGP topology design [17], [33], [34]. However, to the best of our knowledge, few works are specifically targeted to approaches for *modifying* the iBGP configuration of a running network without impacting traffic. Lately, algebraic approaches (e.g., [35]) are also used to analyze routing protocol correctness. We plan to investigate how routing algebras can be used to study the reconfiguration problem in future work.

A general approach to deal with multiple configurations is proposed in [36]. In that work, Alimi *et al.* propose firmware modifications that enable routers to manage a shadow configuration beyond the active configuration used for data traffic forwarding. Shadow and active configurations can be switched using an ad-hoc commit protocol. The entire approach could be seen as a way to implement two BGP control planes. However, our solution is more lightweight and easier to implement with respect to [36], as it requires no device modification, and no ad-hoc protocol for either tagging packets and committing configuration changes. In this paper, we also justified the need for an additional control plane to solve the BGP reconfiguration problem by a thorough theoretical study.

Graceful session reset is tackled in [25]. Also, Route Refresh and BGP Soft-Reset capabilities are standardized in [37]. Contrary to these approaches, we aim at enabling iBGP and eBGP reconfigurations which are not restricted to single BGP peerings and affect several routers in a network.

Recently, some techniques [38], [39] have been proposed to enable virtual routers or parts of the configuration of BGP routers (e.g., BGP sessions) to be moved from one physical device to another. Their works differ from ours as we aim at changing network configurations.

In [40], Reitblat *et al.* study the problem of consistent network updates in software defined networks. They propose a set of consistency properties and show how these properties can be preserved when changes are performed in the network. Unlike our approach, this work only applies to logically-centralized networks (e.g., OpenFlow).

Some recent work addressed network-wide link-state IGP reconfigurations [2], [41]. The main idea consists in simultaneously running two IGPs, and finding an operational ordering in which to reconfigure routers without creating forwarding anomalies. Taking inspiration from these techniques, we proposed to rely on two BGP control planes. However, the algorithms proposed in [2], [41] to avoid packet losses in IGP reconfigurations cannot be extended to BGP, mainly because of the different nature of the protocols. Indeed, contrary to the neat route visibility ensured by link-state IGPs, only best routes are propagated by each BGP router, hence a single change on one BGP router can have unexpected side effects on routing information received by remote routers.

VIII. Conclusions

Network operators regularly change router configurations. BGP reconfigurations do not make an exception, as confirmed by our analysis of a Tier-1 ISP's historical configuration data. Since today's SLAs are stringent, reconfigurations must be performed with minimal impact on data plane traffic.

In this paper, we show that routing and forwarding anomalies, possibly resulting in high packet loss ratios, can occur during BGP reconfigurations, even when MED is not used and simple policies are deployed. Unfortunately, current best practices do incur long-lasting anomalies even during common BGP reconfigurations, as we show by simulating a full-mesh to route reflection reconfiguration on a Tier-1 ISP. Hence, we study the problem of finding an operational ordering so that all intermediate configurations are anomaly-free. Unfortunately, the problem of deciding whether such an ordering exists is computationally intractable. Also, we show several cases where such an ordering simply does not exist. Finally, we propose a solution that enables provably lossless BGP reconfigurations by leveraging existing technology to run multiple isolated control planes in parallel. We describe an implementation of this framework, evaluate its scalability, and illustrate its effectiveness through a case-study.

Our findings show that achieving lossless BGP reconfigurations is a hard problem in the general case. However, there might exist specific reconfigurations that can be performed safely, i.e., without relying on multiple control planes. Understanding what kinds of reconfigurations can be carried out under what assumptions remains an interesting open problem.

References

[1] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, 2006.

[2] L. Vanbever, S. Vissicchio, C. Pelsser, P. Francois, and O. Bonaventure, "Seamless Network-Wide IGP Migrations," in *Proc. SIGCOMM*, 2011.

[3] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS Weights in a Changing World," *IEEE Jour. on Sel. Areas in Comm.*, vol. 20, no. 4, pp. 756 –767, May 2002.

[4] T. Griffin and G. Wilfong, "On the correctness of ibgp configuration," in *Proc. SIGCOMM*, 2002.

[5] S. Vissicchio, L. Cittadini, L. Vanbever, and O. Bonaventure, "i-BGP Deceptions: More Sessions, Fewer Routes," in *Proc. INFOCOM*, 2012.

[6] S. Balon and G. Leduc, "Combined Intra- and Inter-domain Traffic Engineering using Hot-Potato Aware Link Weights Optimization," in *Proc. SIGMETRICS*, 2008.

[7] M. Brown, C. Hepner, and A. Popescu, "Internet captivity and depeering," NANOG 45, 2009.

[8] S. Lee, T. Wong, and H. Kim, "To automate or not to automate: On the complexity of network configuration," in *IEEE ICC 2008*, Beijing, China, May 2008.

[9] P. Smith, "BGP Techniques for Service Providers," NANOG 50, 2010.

[10] G. Herrero and J. van der Ven, *Network Mergers and Migrations: Junos Design and Implementation*. Wiley Publishing, 2010.

[11] R. Zhang and M. Bartell, *BGP Design and Implementation*. Cisco Press, 2003.

[12] "Equinix direct," http://www.equinix.com/solutions/regional-solutions/americas/equinix-direct/.

[13] T. Bates, E. Chen, and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)," RFC 4456, 2006.

[14] N. Feamster and H. Balakrishnan, "Detecting BGP Configuration Faults with Static Analysis," in *Proc. NSDI*, 2005.

[15] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The Stable Paths Problem and Interdomain Routing," *IEEE/ACM Trans. Netw.*, vol. 10, pp. 232–243, April 2002.

[16] T. Griffin and G. T. Wilfong, "Analysis of the MED Oscillation Problem in BGP," in *Proc. ICNP*, 2002.

[17] M.-O. Buob, S. Uhlig, and M. Meulle, "Designing optimal ibgp route-reflection topologies," in *Proc. Networking*, 2008.

[18] C. Villamizar, R. Chandra, and R. Govindan, "BGP Route Flap Damping," RFC 2439, Internet Engineering Task Force, 1998.

[19] S. Vissicchio, "Governing Routing in the Evolving Internet," PhD. Thesis, 2012, http://www.dia.uniroma3.it/~compunet/www/docs/vissicchio-thesis-text.pdf.

[20] B. Quoitin and S. Uhlig, "Modeling the Routing of an Autonomous System with C-BGP," *IEEE Network*, vol. 19, no. 6, November 2005.

[21] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet Inter-Domain Traffic," in *Proc. SIGCOMM*, 2010.

[22] S. Uhlig and S. Tandel, "Quantifying the BGP routes diversity inside a tier-1 network," in *Proc. Networking*, 2006.

[23] "Configuring virtual routing and forwarding," Official Cisco Documentation, Cisco Systems, Inc., www.cisco.com.

[24] V. Valancius and N. Feamster, "Multiplexing BGP sessions with BGP-Mux," in *Proc. CoNEXT*, 2007.

[25] P. Francois, B. Decraene, C. Pelsser, K. Patel, and C. Filsfils, "Graceful BGP session shutdown," Internet-Draft, December 2011.

[26] "GEANT Backbone Topology," 2011, http://www.geant.net.

[27] "Seamless BGP Reconfigurations," http://inl.info.ucl.ac.be/softwares/, 2012.

[28] H. Ballani, P. Francis, T. Cao, and J. Wang, "Making routers last longer with ViAggre," in *Proc. NSDI*, 2009.

[29] RIPE Routing Information Service (RIS), http://www.ripe.net/ris.

[30] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, "Design and implementation of a routing control platform," in *Proc. NSDI*, 2005.

[31] R. Chen, A. Shaikh, J. Wang, and P. Francis, "Address-based route reflection," in *Proc. CoNEXT*, 2011.

[32] R. Mahajan, D. Wetherall, and T. Anderson, "Understanding BGP misconfiguration," in *Proc. SIGCOMM*, 2002.

[33] A. Rawat and M. A. Shayman, "Preventing Persistent Oscillations and Loops in IBGP Configuration with Route Reflection," *Comput. Netw.*, vol. 50, pp. 3642–3665, December 2006.

[34] M. Vutukuru, P. Valiant, S. Kopparty, and H. Balakrishnan, "How to Construct a Correct and Scalable iBGP Configuration," in *Proc. INFOCOM*, 2006.

[35] T. Griffin and J. L. Sobrinho, "Metarouting," in *Proc. SIGCOMM*, 2005.

[36] R. Alimi, Y. Wang, and Y. R. Yang, "Shadow configuration as a network management primitive," in *Proc. SIGCOMM*, 2008.

[37] E. Chen, "Route Refresh Capability for BGP-4," RFC 2918, 2000.

[38] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe, and J. Rexford, "Virtual routers on the move: live router migration as a network-management primitive," in *Proc. SIGCOMM*, 2008.

[39] E. Keller, J. Rexford, and J. Van Der Merwe, "Seamless BGP Migration with Router Grafting," in *Proc. NSDI*, 2010.

[40] M. Reitblatt, N. Foster, J. Rexford, and D. Walker, "Consistent Updates for Software-Defined Networks: Change You Can Believe In!" in *Proc. HotNets-X*, 2011.

[41] S. Raza, Y. Zhu, and C.-N. Chuah, "Graceful Network State Migrations," *Trans. on Netw.*, vol. 19, no. 4, pp. 1097 –1110, 2011.

Stefano Vissicchio received his Master degree in computer science from the Roma Tre University in 2008, and the Ph.D. degree from the same institution in April 2012. He collaborated with Consortium GARR between 2011 and 2012. Currently, he has a post-doctorate position at the Université catholique de Louvain. His research interests are mainly focused on network management and routing. He is currently working on routing protocol configuration design, testing, and deployment.

Laurent Vanbever is currently finishing his Ph.D. at the Université catholique de Louvain. He obtained his Master degree in computer science from the Université catholique de Louvain in 2008. His research interests focus on network management, notably IP routing, network configuration management, and network validation. Laurent Vanbever also holds a master degree in business management from the Solvay Brussels School of Economics and Management.

Cristel Pelsser received her Master degree in computer science from the FUNDP in Belgium in 2001. She then obtained her PhD in applied sciences from the UCL, in Belgium, in 2006. From 2007 to 2009, she held a post-doctorate position at NTT Network Service Systems Laboratories in Japan. She is now a researcher at Internet Initiative Japan (IIJ). Her current research interests are in Internet routing and privacy in distributed storage systems.

Luca Cittadini received his Master degree from the Roma Tre University in 2006, and a Ph.D. degree in Computer Science and Automation from the same institution in 2010, defending the Thesis "Understanding and Detecting BGP Instabilities". During his Ph.D. he was a teaching assistant in the computer network research lab. His research activity is primarily focused on interdomain routing, including theoretical analysis of the protocol as well as active and passive measurements.

Pierre Francois is a Staff Researcher at Institute IMDEA Networks since September 2011. He received his Master degree in Computer Science from the Facultés Notre Dame de la Paix in Namur, and he obtained his Ph.D. degree from Université catholique de Louvain in 2007. He received the IEEE INFOCOM 2007 Best Paper Award. His main topics of interest are notably IP Routing scaling and convergence, Internet governance, Internet routing economics, and network measurements. Pierre Francois is also active in standardization, holding an extensive list of IETF contributions.

Olivier Bonaventure graduated from the University of Liège in 1992. He obtained his Ph.D. degree in 1999 and spent one year at Alcatel in Antwerp. He is now full Professor at Université catholique de Louvain, where he leads the IP Networking Lab (http://inl.info.ucl.ac.be) and is Vice-President of the ICTEAM Institute. He has published more than eighty papers, contributes to IETF, was granted several patents and served on the editorial board of IEEE/ACM Transactions on Networking. He currently serves as Education Director within ACM SIGCOMM, and is a member of the CoNEXT steering committee. He received several awards including the INFOCOM 2005 best paper award.