On the Co-Existence of Distributed and Centralized Routing Control-Planes

Stefano Vissicchio*, Luca Cittadini‡, Olivier Bonaventure*, Geoffrey G. Xie§, Laurent Vanbever†
*Université catholique de Louvain, ‡RomaTre University, §Naval Postgraduate School, †Princeton University

***Abstract*—Network operators can and do deploy multiple routing control-planes, e.g., by running different protocols or instances of the same protocol. With the rise of SDN, multiple control-planes are likely to become even more popular, e.g., to enable hybrid SDN or multi-controller deployments. Unfortunately, previous works do not apply to arbitrary combinations of centralized and distributed control-planes.**

In this paper, we develop a general theory for coexisting control-planes. We provide a novel, exhaustive classification of existing and future control-planes (e.g., OSPF, EIGRP, and OpenFlow) based on fundamental control-plane properties that we identify. Our properties are general enough to study centralized and distributed control-planes under a common framework. We show that multiple uncoordinated control-planes can cause forwarding anomalies whose type solely depends on the identified properties. To show the wide applicability of our framework, we leverage our theoretical insight to (i) provide sufficient conditions to avoid anomalies, (ii) propose configuration guidelines, and (iii) define a provably-safe procedure for reconfigurations from any (combination of) control-planes to any other. Finally, we discuss prominent consequences of our findings on the deployment of new paradigms (notably, SDN) and previous research works.

I. Introduction

Intradomain routing is key to network operation. Luckily, operators have several degrees of control on it. They can choose from a variety of routing protocols, including static routing, several Interior Gateway Protocols (IGPs) (e.g., EIGRP, OSPF, or IS-IS), and Software Defined Networking (SDN) ones (e.g., OpenFlow [1]). Each protocol provides configuration knobs (e.g., IGP link weights) to influence route dissemination and forwarding path computation. Also, routers can build multiple *control-planes*, by simultaneously running multiple protocols (or multiple instances of the same protocol, as in OSPF), each with its own configuration, in logically-separated software processes. We say that control-planes are *coexisting* if they run independently from each other *without* exchanging information (e.g., without route redistribution [2]).

As emerged from discussions with Internet Service Provider operators, coexisting control-planes are used in real-world networks for a number of practical use cases. First, multiple control-planes can improve network robustness. For example, they help mitigate the risk of bugs in specific implementations of a given routing protocol by confining them to a single control-plane. Also, if a problem occurs in one control-plane, connectivity can be preserved by shifting traffic to forwarding paths managed by another control-plane. Second, coexisting control-planes can be used for traffic engineering, e.g., assigning distinct classes of traffic to different control-planes. For example, latency-sensitive traffic can be assigned to an IGP control-plane supporting specific traffic engineering features, while best-effort traffic can be handled by a separate control-plane with a less resource-demanding IGP. Third, coexisting control-planes can help accommodate network dynamics, e.g., failures or traffic shifts. For instance, they can improve fast failure recovery [3], and facilitate disruption-free reconfigurations, to arbitrarily change (e.g., for traffic engineering) the configuration of a given protocol (e.g., a link-state IGP [4]) or to migrate from one protocol to another (e.g., [5], [6]).

We expect coexisting control-planes to become even more popular with the growing interest in SDN. First, hybrid SDN networks, running both SDN and traditional routing protocols and enabled by hybrid routers [1], can (*i*) improve routing flexibility with respect to pure IGP networks [7], (*ii*) enable deployment of advanced network capabilities (like network function virtualization) [8], (*iii*) combine the flexibility of SDN with the scalability and robustness of IGPs [9], and (*iv*) enable a smooth migration to a pure SDN deployment [10]. Second, coexisting SDN control-planes capture the case of multiple uncoordinated controllers managing the same devices.

Control-plane coexistence does not create routing anomalies, because no information is exchanged between control-planes. Still, even if each control-plane is correct in the absence of others, inconsistent forwarding entries installed on routers may result in data-plane disruptions.

In this paper, we develop the first theoretical framework to reason about the coexistence of arbitrary control-planes. Our contribution is manifold.

First, in Sec. II, we propose a model for hybrid routers that is general enough to capture heterogeneous control-planes, independent of the adopted path computation algorithms and of the header fields used to match and forward packets.

Second, in Sec. III, we characterize coexisting control-planes prone to forwarding anomalies. We classify control-planes according to two fundamental properties, based on the input data structure (RIB or FIB) from which routes are fetched before being disseminated, and the output data structure (RIB or FIB) where routes are installed. Our classification is (*i*) exhaustive, i.e., it covers existing protocols and future ones applying to the current router design; and (*ii*) novel, as it is orthogonal to traditional classifications like link-state vs. distance-vector protocols. We prove, in Sec. IV, that the kinds of anomalies resulting from control-plane coexistence

Stefano Vissicchio is a postdoctoral researcher of the Belgian fund for scientific research (F.R.S.-FNRS)

depend solely on our classification. For the combinations of control-planes that are not inherently anomaly-free, we provide sufficient conditions that guarantee correctness.

Third, in Sec. V, we exemplify the wide applicability of our theoretical framework by leveraging it to (*i*) propose configuration guidelines that prevent anomalies for coexisting control-planes; and (*ii*) devise a procedure for safe reconfigurations from any combination of control-planes to any other.

Fourth, in Sec. VI, we discuss the implications of our results from the point of view of network operators and protocol designers. Further, we analyze the impact of our findings on the deployment of new protocols and paradigms, focusing on SDN. Notably, our theory exposes behavioral differences of distinct SDN proposals, e.g., showing that a straightforward implementation of OpenFlow is not inherently safe when coexisting with traditional routing protocols. In comparison, competing SDN proposals (e.g., I2RS [11]) provide more correctness guarantees. Our findings also enable us to evaluate risks and consequences of design choices in hybrid SDN networks, like the simultaneous usage of routes provided by OpenFlow and IGP for multi-path routing.

Fifth, in Sec. VII, we discuss related work. Our results generalize and extend previous contributions on safe coexistence of multiple IGP instances (e.g., [12]), hybrid SDN networks (e.g., [7]) and safe reconfigurations (e.g., [4]).

Finally, we conclude in Sec. VIII.

II. Model

In this section, we present our router model and notation. We first describe our model (Sec. II-A), then we formalize the notion of correctness (Sec. II-B), and lastly we discuss the generality of our formalization (Sec. II-C).

A. Routers, Protocols, and Control-Planes

In a network, data packets produced by end hosts are relayed hop-by-hop by intermediate nodes that we call *routers*. Our router model is illustrated in Fig. 1. We defer the discussion of inputs and outputs to Sec. III.

Each router maintains a table called Forwarding Information Base (FIB). For any router r and any destination d, a FIB entry $fib(r,d)$ contains the next-hop of r to d. For any destination d to be reachable, at least one router r must be *directly connected* to d. We refer to the software processes that populate routers' FIB, e.g., by running routing protocols in a given configuration, as *control-planes*. Each control-plane stores its own routing information in a separate Routing Information Base (RIB). We write $rib_M(r,d)$ to indicate the route (i.e., a path on the network) in the RIB of a router r for control-plane M and destination d. If a control-plane M does not provide r with any route for d, then $rib_M(r,d) = \emptyset$.

Routers can run multiple control-planes at the same time. To choose which control-plane writes to the FIB, routers rely on a local *control-plane selection* process. This process is based on the *preference* locally assigned to each control-plane. For example, the preference of IGP control-planes is based on the value of the so-called Administrative Distance (AD)

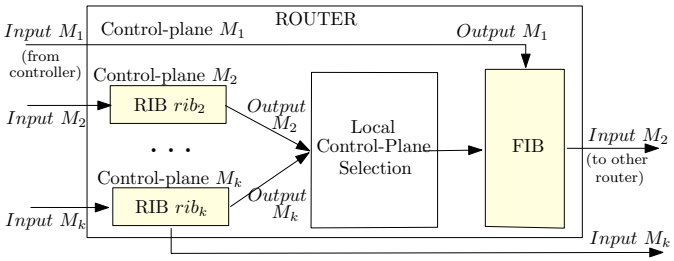

Fig. 1. Router model.

assigned to the corresponding IGPs. Preferences (e.g., AD values) can be typically set by router configuration, on a per-destination basis. Finally, routers select the next-hop from the most preferred control-plane providing a non-empty RIB entry, and copy it to the FIB. We say that a given control-plane M is *used* by a router r for a destination d if M is the control-plane that populates the FIB with an entry $fib(r,d) \neq \emptyset$. We denote the used control-plane as $used(r,d)$. By definition, $M = used(r,d)$ implies that $fib(r,d)$ is the next-hop of r in $rib_M(r,d)$ and $rib_M(r,d) \neq \emptyset$. For directly connected destinations, the control-plane selection process is skipped, and we say that all control-planes are used at the same time.

B. Forwarding Correctness

In this paper, we are interested in studying the impact of control-plane coexistence on routing and forwarding correctness. In the absence of information exchange between control-planes (e.g., route redistribution), routing is guaranteed to be stable [12]. Hence, we focus on forwarding correctness.

We say that a network is *forwarding correct* (or simply *correct*) if for every router r and destination d, the forwarding path from r to d terminates in d. A forwarding path is the concatenation of FIB entries for the same destination. More precisely, a *forwarding path* $\pi(r,d)$ from router r to destination d is a sequence of routers $(v_0 \dots v_k)$ such that $k \geq 0$, $v_0 = r$, and $\forall i = 0, \dots, k-1$, $v_{i+1} = fib(v_i, d)$.

In the case of incorrectness, we distinguish between two *forwarding anomalies*: blackholes and forwarding loops. A *blackhole* at router r for destination d occurs if $\pi(r,d) = (r\ v_0 \dots v_k)$, with $k \geq 0$, $v_k \neq d$ and $fib(v_k, d) = \emptyset$. A *forwarding loop* occurs if $\pi(r,d)$ contains repeated nodes. A forwarding loop directly leads to packet losses since packets are forwarded indefinitely along the loop and eventually discarded. Conversely, when a blackhole exists for a destination d, packets destined to d will be either dropped or forwarded based on a less specific destination (e.g., a default route). In the latter case, packets to the same destination are routed inconsistently among different routers, possibly leading to hard-to-debug forwarding inconsistencies and service disruption.

To focus on forwarding anomalies caused by control-plane coexistence, we assume *correctness in isolation*, i.e., each control-plane is assumed to be stable and forwarding correct in the absence of other control-planes. Routing stability is needed to even define the forwarding state. Moreover, for every control-plane M, we assume that (*i*) no blackhole occurs if M

is the only deployed control-plane, and (*ii*) irrespective of the presence of other control-planes, the RIB entries provided by M to all routers for any given destination never form a loop, which prevents forwarding loops. Distributed routing protocols are correct in isolation by design. For centralized (e.g., SDN) control-planes, we assume correctness to be ensured by the implementation of the controller.

C. Generality of the Model

So far, we have not yet specified what a destination is. We define a *destination* more generally as *any combination of fields* in an IP packet that can be used by a router to select a route. A few examples follow. Any IPv4 or IPv6 prefix is a destination, as routers support destination-based forwarding. Moreover, if per-flow Equal Cost Multi Path (ECMP) is configured, then any source and destination IP pair can be a distinct destination. Similarly, if routing is based on sources and DSCP codepoints, a destination is a combination of DSCP codepoint, source and destination IP prefix.

Control-plane coexistence intrinsically poses two constraints: (*i*) A router must be either directly connected to a destination in all control-planes, or not directly connected to that destination in any control-plane; and (*ii*) the hierarchy of destinations (e.g., the deaggregation of destination IP prefixes) must be consistent across all control-planes. The first constraint is needed to let all control-planes share the same view of the physical topology. The second constraint is needed for control-plane preference to be well-defined, as comparison between control-planes is defined per-destination, Note, however, that this constraint can always be enforced by creating a mapping between destinations used in different control-planes. For example, if an OSPF and an OpenFlow control-planes respectively match destination and source IP prefixes, destination consistency can be enforced if OpenFlow matches both source and OSPF-matched destination IP prefixes (instead of sources only).

III. Control-Plane Taxonomy

We now present our novel control-plane taxonomy. It is based on fundamental control-plane properties presented in Sec. III-A. Those properties are orthogonal, hence any combination of them maps to a different class of control-planes in our taxonomy. We show how our taxonomy applies to control-planes running existing routing protocols in Sec. III-B.

A. Fundamental Properties of Routing Control-Planes

The properties that characterize our taxonomy relate to control-plane input and output data structures.

Input (Route Dissemination). We distinguish between FIB-Unaware (FU) and FIB-Aware (FA) control-planes. If the FIB is used as an input to route dissemination, we say that the control-plane is FA, otherwise it is FU.

Namely, FU control-planes disseminate the same routes independently of FIB entries, e.g., solely on the basis of the content of the RIB (as M_k in Fig. 1). Consider for example the network topology depicted in Fig. 2, where r_1, r_2 and r_3 are

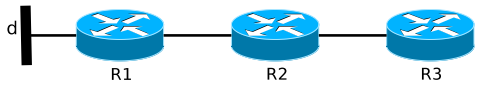

Fig. 2. A simple network to illustrate the difference between FU and FA.

routers, all participating to an FU control-plane M, and d is a destination directly connected to r_1. All routers participate to an FU control-plane M, and $rib_M(r_3, d) = (r_3\ r_2\ r_1\ d)$. Even if r_2's FIB entry for d is provided by another control-plane (e.g., a static route), this does not affect $rib_M(r_3, d)$. That is, the following property holds by definition of FU control-plane.

Property 1: Let M be an FU control-plane. For any router r and destination d, $rib_M(r, d)$ does not depend on any other coexisting control-plane $M' \neq M$.

Conversely, FA control-planes react to FIB changes by distributing updated routes (as M_2 in Fig. 1). In Fig. 2, if the configured control-plane M is FA, then changes to r_2's FIB entry for d cause M to update the RIB entry for d at r_3. For example, if a route from another control-plane (e.g., a static route) is installed in r_2's FIB, then r_2 stops propagating to r_3 the route given by M. More generally, in an FA control-plane, a router propagates a route to a destination d only if that route is used to compute its FIB entry to d. More formally, the following property holds.

Property 2: Let M be an FA control-plane. For any router r and destination d, $rib_M(r, d) = (r\ i \ldots d) \Rightarrow M = used(i, d)$.

Output (Route Installation) We distinguish between NON-PREEMPTIVE and PREEMPTIVE control-planes, depending on whether they output routes directly to the FIB or to the RIB. Traditional control-planes, e.g., running IGPs, are non-preemptive, since they use the routers' RIBs to store their respective best routes (as M_2 and M_k in Fig. 1). On the contrary, SDN control-planes, e.g., running OpenFlow, typically move routing information out of routers. Routes are indeed computed and stored in a logically-centralized controller, which push them directly to router FIBs (as M_1 in Fig. 1). Hence, OpenFlow control-planes are preemptive.

Despite the fact that preemptive control-planes bypass the control-plane selection process, they still allow network operators to configure per-destination control-plane preference. For example, OpenFlow switches can use the so-called "normal port" to defer the forwarding decision to other control-planes (see Section 5.1 of [1]). However, since the control-plane selection process is bypassed, it is not possible for a non-preemptive control-plane to defer the forwarding decision to a preemptive one. We model this asymmetry by imposing that, for any router r, a preemptive control-plane either (i) is the most preferred one and used by r, or (ii) does not provide any route. More formally, the following property holds.

Property 3: Let M be preemptive control-plane. For any router r and destination d, $rib_M(r, d) \neq \emptyset$ if and only if M is the most preferred control-plane by r.

If several preemptive control-planes are present (e.g., two uncoordinated SDN controllers), the most preferred control-

control-plane	Properties
OpenFlow, ForCES	preemptive, FU
Static routes, RCP, I2RS	non-preemptive, FU
OSPF, ISIS, BGP-as-IGP	non-preemptive, FU
RIP, EIGRP	non-preemptive, FA

TABLE I
CLASSIFICATION OF CONTROL-PLANES RUNNING EXISTING PROTOCOLS.

plane by any router r is the last one that wrote to r's FIB.

B. Mapping Properties to Routing Control-Planes

Our taxonomy is *general* enough to capture a wide variety of control-planes, and expose their differences. Table I reports the classification, according to commercial (Cisco and Juniper) implementations, of the currently most popular control-planes.

Different IGP-based control-planes belong to distinct classes. While all are non-preemptive, OSPF and ISIS build FU control-planes, while RIP and EIGRP lead to FA ones. Furthermore, BGP when used for intradomain routing[1] [13] behaves as an FU control-plane.

We *experimentally verified* those claims by simulating the network depicted in Fig. 2 with Cisco routers. In particular, for each routing protocol, we set up a distinct experiment. In each experiment, we configured r_1, r_2 and r_3 to talk a given protocol, and we continuously probed d from r_3. After checking the correctness of the basic setup, we added a static route for d on r_2, such that the static route was preferred over the IGP one. With RIP and EIGRP, the probes started failing. In fact, the static route caused the r_2's FIB to be updated and r_2 to send a route withdrawal to r_3, consistently with Property 2. Conversely, in OSPF, ISIS and BGP, the static route did not impact the ability of r_3 to reach d.

Similarly, SDN control-planes like those built on RCP [14], I2RS [11], ForCES [15] and OpenFlow [16] also fall in distinct classes. All of them are FU, at least in their basic configuration in which the SDN controller takes its routing decisions independently of the content of routers' FIBs. However, only RCP and I2RS are non-preemptive, since ForCES and OpenFlow control-planes write to the routers' FIBs.

Note that our taxonomy is *orthogonal to traditional classifications*, e.g., between distance-vector and link-state protocols. For instance, EIGRP and BGP are both distance-vector protocols, yet they are in distinct classes.

Even more interestingly, our taxonomy is also *exhaustive*. Indeed, as long as routers can be represented by our model (e.g., see Fig. 1), a control-plane must be either FU or FA (i.e., reacting to FIB changes or not), and either preemptive or non-preemptive (i.e., writing to the FIB or to the RIB). Abstracting away all internal details and restricting to the analysis of their input/output properties allows us to model future control-planes. For instance, we can model any FU version of distance-vector IGPs, as well as FA variants of OpenFlow. We leverage this generality to evaluate different proposals for SDN protocol implementation in Sec. VI.

[1]since we focus on intradomain routing, we do not consider the usage of BGP for interdomain routing

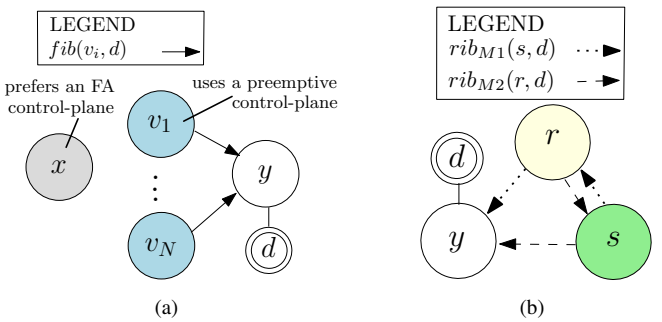

Fig. 3. Abstract examples showing that (a) blackholes can occur if only preemptive and FA control-planes coexist, and (b) forwarding loops can occur in the presence of multiple FU control-planes. Circles represent routers, while the double-circled node identifies a destination d.

IV. CHARACTERIZATION OF FORWARDING ANOMALIES WITH COEXISTING CONTROL-PLANES

In this section, we show that the properties identified in Sec. III determine correctness guarantees of arbitrary coexisting control-planes (Sec. IV-A). We also provide sufficient conditions to avoid forwarding anomalies in potentially troublesome combinations of coexisting control-planes (Sec. IV-B).

A. Taxonomy-Based Characterization

We start by dealing with blackholes, i.e., identifying all the combinations of control-planes in which blackholes can occur. The following theorem holds.

Theorem 1: Coexisting control-planes are guaranteed to be free of blackholes if and only if they i) include non-preemptive FU control-planes, or ii) do not include both a preemptive control-plane M and an FA control-plane $M' \neq M$.

Proof: We prove the statement in two steps.

If coexisting control-planes do not comply with Conditions i) and ii), then blackholes can occur. Consider the example in Fig. 3(a), in which circles represent routers and d is a destination attached to y. An FA control-plane is the most preferred one by x, while all neighbors $v_1, \ldots, v_N$ of x prefer and use a preemptive control-plane. Consider the RIB entries of x towards d. To not violate Property 2, $rib_M(x, d) = \emptyset$ for all FA control-planes M running on x. Also, by Property 3, $rib_{\bar{M}}(x, d) = \emptyset$ for all preemptive control-planes $\bar{M}$ running on x. By hypothesis, there are no non-preemptive FU control-planes. Hence, x cannot use any control-plane to reach d, which creates a blackhole $\pi(x, d) = (x)$.

If coexisting control-planes comply with Condition i) or ii), then blackholes cannot occur. If Condition i) is satisfied, then a non-preemptive FU control-plane M exists. By Property 1 and the assumption of correctness in isolation, $rib_M(r, d) \neq \emptyset$ for any router r and destination d, irrespectively of the presence of other control-planes. Thus, any router can always rely on M to populate its FIB to any destination, which guarantees the absence of blackholes.

Otherwise, if Condition i) does not hold but Condition ii) does, we have two cases: either all control-planes are preemptive FU, or they are all non-preemptive FA.

In the former case, for every destination d, every node r must prefer one preemptive FU control-plane M_r. M_r provides r with a route to d by Property 3, hence r uses M_r. In other words, each node has a FIB entry for every destination, which guarantees the absence of blackholes.

In the latter case, all control-planes are non-preemptive FA. Assume by contradiction that a blackhole exists for router r and destination d. Since all control-planes are non-preemptive, it must be $rib_M(r,d) = \emptyset$ for every control-plane M. By Property 2, this implies $rib_M(r',d) = \emptyset$ for every control-plane M and every neighbor r' of r. By iterating the same argument, we eventually conclude that for every router, every control-plane does not provide any route for d. This must also hold for the routers directly connected to d, which contradicts the assumption of correctness in isolation. ■

We now characterize the control-plane combinations which are prone to forwarding loops. To this end, we leverage the following lemma.

Lemma 1: For any router r and destination d, if $used(r,d)$ is an FA control-plane M, $\forall v_i \in \pi(r,d)$ such that v_i is not directly connected to d, $used(v_i,d) = M$.

Proof: Let $\pi(r,d) = (r\ x)P$. If x is directly connected to d, then P is empty, and the statement directly follows. Otherwise, x must use M by Property 2. Thus, we can repeat the same argument to $\pi(x,d)$. The statement follows by noting that we eventually reach a router directly connected to d. ■

Lemma 1 helps us prove the following theorem.

Theorem 2: A combination of routing control-planes is guaranteed to be free of forwarding loops if and only if it includes at most one FU control-plane.

Proof: We prove the statement in two steps.

If at least two least FU *control-planes coexist, then forwarding loops can occur.* Consider the example in Fig. 3(b), with d being a destination. Let $used(r,d) = M_1$ and $used(s,d) = M_2$, with $M_2 \neq M_1$, as highlighted in the figure by the different fillings of r and s. The route towards d that M_1 (M_2, resp.) provides to r (s, resp.) in the absence of other control-planes is via s (r, resp.). By Property 1, those routes are in the respective RIBs of r and s independently of the presence of other coexisting control-planes. Because of control-plane preferences, they are also used to fill the FIB of r and s which creates a forwarding loop $\pi(s,d) = (s\ r\ s)$.

If at most one FU *control-plane coexist with any number of* FA *ones, then forwarding loops cannot occur.* Consider any router r and any destination d. We have three cases. If $used(r,d) = \emptyset$, then there is a blackhole $\pi(r,d) = (r)$, hence no forwarding loop can involve r. Otherwise, if $used(r,d)$ is an FA control-plane M, then Lemma 1 ensures that all the routers in $\pi(r,d)$ use M, and the absence of forwarding loops follows by the correctness in isolation of M. Finally, if $used(r,d)$ is an FU control-plane M', then let $rib_{M'}(r,d) = PQ$, with P including only routers (at least r) using M', and Q the (possibly empty) remaining path. The absence of forwarding loops in P is ensured by the combination of Property 1 and the correctness in isolation of M'. Hence, if Q is empty, the statement directly follows. Otherwise, the forwarding loop must be found in Q. Let x be the first router in Q. Since, by hypothesis, M' is the only FU control-plane, $used(x,d)$ is an FA control-plane. We can then apply one of the two previous arguments to x, proving the statement. ■

M_1 \ M_2	p, FA	p, FU	n, FA	n, FU
p, FA	B	B	B	-
p, FU	B	L	B	L
n, FA	B	B	-	-
n, FU	-	L	-	L

LEGEND: B=blackholes, L=forwarding loops

TABLE II
CHARACTERIZATION OF FORWARDING ANOMALIES FOR TWO COEXISTING CONTROL-PLANES. THEOREMS 1 AND 2 GENERALIZE THESE RESULTS TO AN ARBITRARY NUMBER OF CONTROL-PLANES.

Together, Theorems 1 and 2 characterize the kinds of forwarding anomalies that can occur for any combination of an arbitrary number of coexisting control-planes. As an example, Table II shows the anomaly characterization deriving from those theorems in the case of two coexisting control-planes. In the table, p and n respectively stand for preemptive and non-preemptive. Interestingly, blackholes and forwarding loops never happen at the same time. Also, blackholes can exist only in the presence of a preemptive control-plane (e.g., OpenFlow), and that forwarding correctness is guaranteed for combinations of FU and FA IGPs (which are non-preemptive by definition). We further discuss the implications of our theoretical findings on (hybrid) SDN networks in Sec. VI.

B. Sufficient Conditions for Anomaly-Prone Combinations

We now propose sufficient conditions to guarantee forwarding correctness of control-plane combinations which do not comply with Theorems 1 and 2. We refer to those combinations as *blackhole-prone* and *loop-prone* respectively.

First, we introduce a sufficient condition to prevent blackholes in blackhole-prone control-plane combinations. To this end, we need to introduce a few extra concepts. Given a network graph G, a control-plane M and a destination d, we denote the set of connected components of G containing only routers preferring M for d as $\Gamma(G,M,d)$. Also, we say that a connected component C is *attached* to a destination d if any router $r \in C$ or a neighbor of r is directly connected to d.

Lemma 2: Let r be a router, r' be a neighbor of r, and d a destination directly connected to r. Then, $used(r',d) = M$, with M being the most preferred control-plane of r' for d.

Proof: Since d is directly connected to r, r uses all the coexisting control-planes for d. In particular, it uses M, and makes the route $(r'\ r\ d)$ available to r' in M. Hence, $rib_M(r',d) \neq \emptyset$. Since M is the most preferred control-plane by r' for d by hypothesis, r' uses M to populate its FIB, yielding the statement. ■

Intuitively, we can use Lemma 2 to prove that, in a connected component $C \in \Gamma(G,M,d)$ attached to destination d, at least one router is guaranteed to use control-plane M. This enables us to prove the following theorem.

Theorem 3: If for every destination d, each router belongs to a connected component $C \in \Gamma(G, M, d)$ for some control-plane M, such that C is attached to d, blackholes cannot occur.

Proof: Let d be a destination, and $C \in \Gamma(G, M, d)$ be a connected component attached to d, for some control-plane M. We now show that all the routers in C use M.

If M is FU, it directly follows by noting that all routers in C prefer M (by definition of C) and $\forall r \in C\ rib_M(r, d) \neq \emptyset$ (because of Property 1, possibly combined with Property 3 for preemptive control-planes). Otherwise, if M is FA, let r be any router in C. By definition of C, a path $(v_0, \ldots, v_k)$, with $k \geq 0$, must exist such that $v_0 = r$, $\forall i = 0, \ldots, k\ v_i \in C$, and v_k is either directly connected to d or is a neighbor of a router (possibly not in C) directly connected to d. All routers v_i prefer M, by definition of C. Moreover, Lemma 2 implies $used(v_k, d) = M$. By Property 2, we derive $rib_M(v_{k-1}, d) \neq \emptyset$, hence $used(v_{k-1}, d) = M$ because M is the most preferred control-planes at v_{k-1}. By iterating the same argument on all v_i routers, we eventually conclude $\forall r \in C\ used(r, d) = M$.

Since every router $r \in C$ uses M, then $fib(r, d) \neq \emptyset$, which prevents blackholes inside C. The statement then follows by noting that, by hypothesis, for every router r and every destination d, r belongs to some $C \in \Gamma(G, M, d)$ for some control-plane M, with C attached to d. ∎

We now deal with forwarding loops. To this end, we define the *preferred* FU *graph* to a destination d as the graph obtained by merging the RIB entries of the most preferred FU control-plane of every router in the network. More formally, given a destination d, the preferred FU graph to d contains all and only the edges corresponding to $rib_{M^*}(r, d)$, where r is a router and M^* is the most preferred among the FU control-planes at r. Observe that M^* is not necessarily the most preferred control-plane at r, hence the preferred FU graph do not coincide in general with the graph we would obtain by simply merging all the FIB entries.

Theorem 4: If for every destination d the preferred FU graph is acyclic, then no forwarding loop can occur.

Proof: Assume by contradiction that a forwarding loop L exists for a destination d. Let r be a router in L, i.e., $\pi(r, d) = (l_0\ l_1 \ldots l_k\ l_0)$ with $k \geq 1$ and $l_0 = r$. Let $M = used(r, d)$.

If M is FA, $\pi(r, d)$ contains only routers that use M, by Lemma 1. Hence, L contradicts the assumption of correctness in isolation of M. Thus, M must be FU. To be used, M must actually be the most preferred FU control-plane at r.

By iterating the same argument on all routers in L, we conclude that every $l_i \in L$ use the most preferred FU control-plane at l_i. Hence, the existence of L contradicts the hypothesis that the preferred FU graph is acyclic. ∎

V. Extensions and Applications

In this section, we apply the theoretical insights developed in Sec. IV to devise configuration guidelines (Sec. V-A) and a generic network reconfiguration strategy (Sec. V-B).

A. Configuration Correctness

Sec. IV provides sufficient conditions to guarantee the absence of forwarding anomalies. We leverage them to propose the configuration guidelines in Fig. 4 and 5. A network which is compliant with *any* of Guidelines A1, A2 or A3 is provably free from blackholes by Theorems 1 (for A1 and A2) and 3 (for A3). Similarly, compliance with *either* Guideline B1 or B2 ensures the absence of forwarding loops, by Theorems 2 and 4 respectively.

Guidelines to avoid blackholes

A1: Do not use preemptive control-planes.

A2: Run at least one non-preemptive FU control-plane.

A3: For any destination d and control-plane M, configure at least one neighbor of a router directly connected to d to prefer M over any other control-plane.

Fig. 4. Compliance with any of these guidelines ensures no blackholes.

Guidelines to avoid forwarding loops

B1: Run at most one FU control-plane.

B2: Configure FU control-planes so that, for any destination, their combined routes do not contain loops.

Fig. 5. Compliance with any of these guidelines ensures no forwarding loops.

Guidelines A1, A2, and B1 only limit the classes (according to our taxonomy) of coexisting control-planes, without constraining the relative preference between control-planes or the installed forwarding paths. For these reasons, the guarantees that they provide are *robust to network failures*. Unfortunately, they are not universally applicable. For example, a network operator may specifically need to run multiple FU control-planes (violating B1) or could not run a non-preemptive FU control-plane, e.g., because it is not supported by all routers in her network (violating A2). In these cases, Guidelines A3 and B2 can be followed to guarantee correct forwarding at the cost of restricting control-plane preferences based on the network topology (A3) or constraining the forwarding paths computed by FU control-planes (B2). Guidelines A3 and B2 are not robust to failures. However, they can be used to perform *what-if analyses* about the correctness robustness, e.g., by simulating a number of failures and verifying that they do not affect compliance with the guidelines.

B. Graceful Reconfigurations

We now leverage the theoretical insights described in Sec. IV to study *safe reconfigurations from any combination of coexisting control-planes to any other*.

Live reconfigurations are crucial to adapt to network dynamics, ensure high performance under changing traffic conditions, and improve network flexibility and evolvability [4], [17]. A commonly-used reconfiguration framework, called "Ships in the Night" (SITN) [6], [18], [5], [4], is based on running multiple independent control-planes on the same network. For the sake of simplicity, we now assume the network to run a single control-plane before (and after) the reconfiguration. To change the routing configuration of a network, SITN performs three logical steps: (*i*) introduces the final control-plane (final

1: ***compute_operational_order***(G,M_{init},M_{fin},D)
2: $seq \leftarrow None$
3: **if** the combination of M_i and M_f is blackhole-prone **then**
4: $M_{tmp} \leftarrow$ FU control-plane
5: $seq \leftarrow$ compute_operational_order(G,M_{init},M_{tmp},D) + compute_operational_order(G,M_{tmp},M_{fin},D)
6: **else if** the combination of M_i and M_f is loop-prone **then**
7: $seq \leftarrow$ compute_loopfree_order(G,M_{init},M_{fin},D)
8: **else**
9: $seq \leftarrow$ get_any_order(G)
10: **end if**
11: **return** seq

Fig. 6. A generic forwarding-correct procedure for reconfigurations from any control-plane to any other.

protocol with its final configuration) as the least preferred control-plane on all routers; (*ii*) iteratively changes control-plane preference on a per-router basis, so that the final control-plane gradually becomes the most preferred network-wide; and (*iii*) removes the (no longer used) initial control-plane. As it proceeds with changing the control-plane preference at each router, SITN produces a series of intermediate configurations. Even if the initial and final state are correct, non-transient anomalies can occur in intermediate configurations [4], depending on the applied sequence of operations.

Our theory enables both prediction and prevention of anomalies that can occur during any SITN-based reconfiguration. Indeed, our theoretical results apply to every intermediate configuration generated in the reconfiguration process.

In particular, Theorems 1 and 2, we are able to *predict possible reconfiguration anomalies*, based on just the generic properties of the initial and final control-planes. For example, Table II shows that forwarding loops can occur if and only if both the initial and final control-planes are FU, as in the replacement of one OSPF configuration with another, or in a migration from IS-IS to OpenFlow.

For reconfigurations in which connectivity can be disrupted, we devise a *generic procedure to preserve forwarding correctness* throughout the reconfiguration. Our procedure, summarized in Fig. 6, is based on a static analysis of the control-planes involved in the reconfiguration. Again, we assume for simplicity that a single control-plane is configured before and after the reconfiguration, but *the procedure is easy to extend to the general case of any control-plane combination*.

Basically, we distinguish three cases.

Case 1) If the coexistence of the initial control-plane M_{init} and the final one M_{fin} is blackhole-prone, we split the reconfiguration in two macro-steps (lines 3-5). In the first step, M_{init} is replaced by a temporary FU control-plane M_{tmp}. In the second step, M_{tmp} is replaced by M_{fin}. To perform each step, the procedure is called recursively. The presence of M_{tmp} ensures that the coexisting control-planes in each step are not blackhole-prone (see Theorem 1), and that the recursive call falls in another case.

Case 2) If both M_{init} and M_{fin} are FU, we compute an operational sequence that avoids loops (lines 6-7). Such a sequence has been proved to always exist [4] if we proceed on a per-destination basis. Multiple destinations can be reconfigured together to speed up the process, as long as the resulting operational sequence complies with Theorem 4.

Case 3) If the coexistence of M_{init} and M_{fin} is guaranteed to be forwarding correct (i.e., according to Theorems 1 and 2), we apply an arbitrary operational order (line 9).

To illustrate how this procedure works in practice, we now discuss a few concrete examples.

Migrating a network from RIP to OSPF, or from EIGRP to OSPF [6], falls in the third case of our procedure, hence any operational order can be applied without incurring forwarding disruptions. Experienced practicioners may have witnessed such migrations, with the former being motivated by the scalability limitations of RIP, and the latter by the preference of industry-standard protocols over proprietary ones. Those migrations have been typically carried out using SITN [6], [18] with no service disruptions. This has been perhaps a matter of luck: if RIP or EIGRP were implemented as FU, the same migration strategy would have failed to preserve connectivity.

Other reconfigurations require more care to avoid forwarding anomalies. For instance, migrating from OSPF to IS-IS [4] or changing link weights in OSPF [19] are loop-prone reconfigurations (see Table II) that we can carry out with our procedure. In this case, however, we need to compute an operational order that avoids forwarding loops (e.g., reusing algorithms in [4]). Note that the same procedure can be used to safely modify FIB entries in OpenFlow networks [17], or to replace OSPF with OpenFlow network-wide.

Finally, a reconfiguration from EIGRP to OpenFlow, e.g., to transition to SDN or to deploy a hybrid SDN network, falls in the first case of our procedure. Hence, we split it in two steps. First, we can migrate the network from EIGRP to OSPF, which is provably anomaly-free. Then, we can replace OSPF with OpenFlow as explained above.

VI. Lessons Learned

We now discuss broader implications of our theoretical findings. We organize them as a set of lessons learned on (*i*) protocol design, with a special focus on the timely problem of SDN incremental deployability (Sec. VI-A), (*ii*) network design and protocol selection (Sec. VI-B), and (*iii*) protocol standardization (Sec. VI-C).

A. Design Protocols with Coexistence in Mind

Our results build a theoretical framework for protocol designers to understand the impact of design choices in networks with multiple control-planes. This is especially important for incremental deployability of new protocols and architectures, recently emerging as a major research problem [20].

As an example of the usefulness of our framework, we analyze coexistence properties of OpenFlow. OpenFlow is largely considered the principal SDN protocol. When deployed in isolation, OpenFlow has several advantages, including simplicity (e.g., of controller-device interactions that

are based on a programmatic interface to devices' FIBs), expressiveness (e.g., ability to match arbitrary packet fields), and moderate hardware requirements (which may lead to the reduction of equipment cost in the long-term). Nevertheless, *a straightforward deployment of OpenFlow in hybrid routers can jeopardize the coexistence with other protocols*.

Namely, OpenFlow is FU and preemptive. Sec. IV highlights that both blackholes and forwarding loops can occur in the presence of coexisting control-planes. This is a challenge for operators that plan for the coexistence of OpenFlow and other protocols in the short or medium term, e.g., to support services based on traditional protocols (like MPLS VPNs) [10]. Even worse, the potential for forwarding anomalies can act as disincentive for operators to start the transition to SDN. In comparison, other non-preemptive protocols like I2RS [11] have the advantage of being provably blackhole-free. Note that a non-preemptive variant of OpenFlow installing entries in the RIB rather than in the FIB, would provide the same advantage. Similarly, a recent OpenFlow specification outlines an FA variant of the protocol, which mandates routers to produce flow removal messages when a coexisting control-plane removes a FIB entry (see Section 5.5 of [1]). This would prevent forwarding loops whenever at most one FU control-plane is used (see Theorem 2).

As an additional example of application of our framework, consider the case in which an operator is willing to enable multi-path (i.e., ECMP) across multiple protocols. For example, he may be tempted to enrich OSPF routes with OpenFlow forwarding paths (e.g., surgically violating IGP shortest paths). Thanks to the generality of our model, we can re-apply our theory entirely. For instance, forwarding loops are not prevented in the previous example (assuming OpenFlow to be FU), but our Guideline B2 can be used to avoid them by ECMP configuration.

B. Design Networks with Coexistence in Mind

Choosing a routing protocol (or a combination of them) is non-trivial for an operator: many factors need to be weighed, including cost, expertise, and protocol-specific features. Our results show that coexistence properties also need to be considered at network design time. For instance, one protocol could be preferred to another based on the ease of gracefully reconfiguring it or (partially) replacing it (see Sec. V-B).

We stress that network operators should not pick a routing protocol based solely on the properties described in Sec. III. For example, Table II should not be misread as suggesting the deployment of non-preemptive FA protocols. Pros and cons of each protocol should be, indeed, carefully evaluated. For example, one downside of FA protocols is that, by definition (see Property 2), they do not guarantee network-wide dissemination of routes, if not deployed in isolation. This makes management and troubleshooting much harder in a multi control-plane setting. More in general, our findings suggest that coexisting control-planes impose a fundamental trade-off between correctness guarantees and ease of operation.

C. Define Inputs and Outputs Unambiguously

As discussed in Section IV, the coexistence properties of a control-plane are determined by its inputs and outputs. This suggests a simple yet fundamental recommendation to protocol designers: The inputs and outputs of a routing protocol should be defined unambiguously. In other words, the choice of inputs and outputs should never be left to the implementor.

Unfortunately, this has not always been the case in the past. As an example of vaguely defined inputs, consider the RFC standardizing RIP [21]. When a RIP router needs to send a message to another router it reads routes from its routing table. Quoting [21], "This table has one entry for every destination that is reachable throughout the system operating RIP." Hence, it is totally unclear whether routes should be fetched from the RIP RIB, from the FIB, or from any other intermediate data structure. As a result, different RIP implementations show heterogeneous interpretations of the standard: RIP is FA in Cisco IOS and in Juniper JunOS, while it is FU in the Quagga routing daemon [22]. We experimentally verified this inconsistency in our testbed (see Fig. 2) with IOS version 12.4, JunOS version 10.1, and Quagga version 0.99.10.

Observe that inconsistent implementations of the same protocol are dangerous because they behave differently in a network that employs multiple control-planes. For example, an operator using best practices [6] to replace RIP with OSPF in a Quagga-based network could experience forwarding loops. Even worse, these anomalies can only be exposed in networks with coexisting control-planes. Within a single control-plane, heterogeneous interpretations are perfectly interoperable, which makes those kinds of inconsistencies unlikely to be caught by any interoperability test suite.

VII. Revisiting Related Work

Our results generalize and extend previous research contributions in different areas. We now revisit the state of the art in each of those areas in the light of our contributions.

Safe coexistence of IGP instances. Prior work [12] considered the independent coexistence of multiple link-state IGP instances, with the goal of providing configuration setting that avoid anomalies. Our results extend the ones in [12] in that: (*i*) we consider arbitrary combinations of control-planes, instead of restricting to link-state IGP ones; (*ii*) Theorem 4 generalizes Guideline 1 in [12]; and (*iii*) we do not assume that routers preference is consistent, which for example allows us to reason about reconfiguration scenarios. Also, our work can be used to guide routing protocol selection (see Sec. VI-B), instead of assuming that link-state IGPs must be used.

The problem of guaranteeing stable routing in the presence of route redistribution between control-planes (e.g., [2], [23], [24]) is orthogonal to our work, which is targeted to the case of non-interacting control-planes. Nevertheless, all our results remain valid whenever routing is stable, e.g., with provably safe route redistribution configurations [24].

Hybrid SDN. Recent research contributions have proposed hybrid SDN networks, where an SDN control-plane coexists

with distributed ones. In particular, [7] studies how to improve traffic engineering by dynamically programming OpenFlow-only devices without modifying link-state IGP routers. By Property 3, we can model this scenario by simply imposing that OpenFlow-only (IGP-only, resp.) routers always prefer the OpenFlow (IGP, resp.) control-plane. Theorem 2 proves that this combination is prone to forwarding loops. The techniques described in [7] avoid them by complying with Guideline B2. However, as noted in Sec. V, this guideline is not inherently robust to topology changes, implying that special care is needed to guarantee correctness under network failures if [7] is applied. Conversely, using non-SDN routers only to build forwarding paths between SDN-enabled devices, like in [8], [25], is a more robust approach, which however does not exploit non-SDN router capabilities (using them as switches).

Graceful Reconfigurations. Recent IGP reconfiguration techniques (e.g., [4]) leverage coexisting control-planes to progressively shift from an initial to a final configuration. Sec. V generalizes those techniques by proposing a provably-safe procedure to reconfigure any combination of control-planes to any other. As such, our procedure supports many use cases (including protocol replacement and traffic engineering) across a wide variety of scenarios, ranging from pure IGP networks to pure SDN and hybrid SDN deployments. For example, configuration changes within the same protocol can be safely performed by simply running different protocol instances in two control-planes. This also applies to pure SDN networks, with the initial and final configurations that can be modeled as different control-planes. Note that, contrary to reconfiguration techniques based on packet tagging (e.g., [17]), our approach *never duplicates FIB entries* on SDN devices. Given the cost of the TCAM memories used to implement FIB tables (e.g., to support OpenFlow's arbitrary bitmask matching), avoiding such duplications ensures scalability and may be even needed in some reconfiguration scenarios.

Multiple control-planes also enable route redistribution reconfigurations [26]. Route redistribution is out of the scope of this work. However, note that the reconfiguration procedure presented in Sec. V allows for a generalization of the algorithms described in [26] to networks running protocols other than link-state IGPs.

VIII. Conclusions

Deploying multiple control-planes in a single network is profitable to improve configuration flexibility, traffic engineering, and robustness to failures and implementation bugs.

In this paper, we provide a characterization of the anomalies due to the presence of multiple non-interacting control-planes. Our characterization is based on fundamental properties of control-planes that are generic enough to apply to (*i*) any number of coexisting control-planes, and (*ii*) all existing and possibly future control-planes (both distributed and centralized). By exploiting our theoretical insights, we propose sufficient conditions and configuration guidelines that guarantee the absence of anomalies, and devise a generalized procedure to perform arbitrary routing reconfigurations without interrupting connectivity. Finally, we show the wide applicability of our findings by discussing their impact on (*i*) the design and standardization of routing protocols, (*ii*) the implementation and incremental deployment of new paradigms like SDN, and (*iii*) the trade-offs that operators need to consider when comparing routing protocols.

Acknowledgements

This work has been supported by the ARC grant 13/18-054 from Communauté française de Belgique.

References

[1] "OpenFlow Switch Specification (version 1.3.4)," March 2014. [Online]. Available: https://www.opennetworking.org/
[2] F. Le, G. G. Xie, and H. Zhang, "Understanding route redistribution," in *Proc. ICNP*, 2007.
[3] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing, and O. Lysne, "Fast IP network recovery using multiple routing configurations," in *Proc. INFOCOM*, 2006.
[4] L. Vanbever, S. Vissicchio, C. Pelsser, P. François, and O. Bonaventure, "Lossless migrations of link-state IGPs," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1842–1855, 2012.
[5] G. Herrero and J. van der Ven, *Network Mergers and Migrations: Junos Design and Implementation*. Wiley, 2010.
[6] J. Parks, *Day One: Migrating EIGRP to OSPF*. Juniper Networks Books, 2011, pp. 19–21.
[7] S. Agarwal, M. Kodialam, and T. V. Lakshman, "Traffic engineering in software defined networks," in *Proc. INFOCOM*, 2013.
[8] T. Koponen et al., "Network virtualization in multi-tenant datacenters," in *Proc. NSDI*, 2014.
[9] O. Tilmans and S. Vissicchio, "IGP-as-a-Backup for Robust SDN Networks," in *Proc. CNSM*, 2014.
[10] S. Vissicchio, L. Vanbever, and O. Bonaventure, "Opportunities and research challenges of hybrid software defined networks," *ACM SIGCOMM CCR*, vol. 44, no. 2, pp. 70–75, 2014.
[11] A. Atlas et al., "Interface to the Routing System Framework," Internet Draft, 2013.
[12] F. Le, G. G. Xie, and H. Zhang, "Instability free routing: Beyond one protocol instance," in *Proc. CoNEXT*, 2008.
[13] P. Lapukhov, A. Premji, and J. Mitchell, "Use of BGP for routing in large-scale data centers," Internet Draft, 2014.
[14] 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.
[15] L. Yang et al., "Forwarding and Control Element Separation (ForCES) Framework," RFC 3746, 2004.
[16] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *SIGCOMM CCR*, vol. 38, no. 2, pp. 69–74, 2008.
[17] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker, "Abstractions for network update," in *SIGCOMM*, 2012.
[18] Hewlett-Packard, "Migration from Cisco IGRP and EIGRP to industry-standard OSPF," Technical White Paper, 2012.
[19] P. Francois and O. Bonaventure, "Avoiding transient loops during igp convergence in ip networks," in *Proc. INFOCOM*, 2005.
[20] M. K. Mukerjee, D. Han, S. Seshan, and P. Steenkiste, "Understanding Tradeoffs in Incremental Deployment of New Network Architectures," in *Proc. CoNEXT*, 2013.
[21] G. Malkin, "RIP Version 2," RFC 2453, 1998.
[22] K. Ishiguro et al., "Quagga routing suite," `www.quagga.net`.
[23] F. Le, G. G. Xie, and H. Zhang, "Theory and new primitives for safely connecting routing protocol instances," in *Proc. SIGCOMM*, 2010.
[24] F. Le and G. Xie, "On Guidelines for Safe Route Redistributions," in *Proc. ACM INM*, 2007.
[25] D. Levin, M. Canini, S. Schmid, F. Schaffert, and A. Feldmann, "Panopticon: Reaping the Benefits of Incremental SDN Deployment in Enterprise Networks," in *Proc. USENIX ATC*, 2014.
[26] S. Vissicchio, L. Vanbever, L. Cittadini, G. Xie, and O. Bonaventure, "Safe Routing Reconfigurations with Route Redistribution," in *INFOCOM*, 2014.