Formal Semantics and Automated Verification for the Border Gateway Protocol

Konstantin Weitz weitzkon@cs.uw.edu
Doug Woos dwoos@cs.uw.edu
Emina Torlak emina@cs.uw.edu
Michael D. Ernst mernst@cs.uw.edu
Arvind Krishnamurthy arvind@cs.uw.edu
Zachary Tatlock ztatlock@cs.uw.edu

University of Washington

Traffic is routed across the Internet by Autonomous Systems, or ASes, such as ISPs, corporations, and universities. To route traffic reliably and securely, ASes must configure their Border Gateway Protocol (BGP) routers to implement policies restricting how routing announcements can be used and exchanged with other ASes.

It is challenging to correctly implement BGP policies in low-level configuration languages. Large ASes maintain millions of lines of frequently-changing configurations that run distributed across hundreds of routers [8, 16]. Router misconfigurations are common and have led to highly visible failures affecting ASes and their billions of users. For example, in 2009 YouTube was inaccessible worldwide for several hours due to a misconfiguration in Pakistan [2], and in 2010 and 2014 China Telecom hijacked significant but unknown fractions of international traffic for extended periods [4, 15, 11, 10]. Goldberg surveys several additional major outages and their causes [7].

We present the first mechanized formal semantics of the BGP specification RFC 4271 [14], and we show how to use this semantics to develop reliable tools and guidelines that help BGP administrators avoid router misconfiguration. In contrast to previous semantics [6, 3, 17], our semantics is fully formal (it is implemented in the Coq proof assistant), and it models all required features of the BGP specification modulo low-level details such as bit representation of update messages and TCP.

To provide evidence for the correctness and usefulness of our semantics: 1) we have extended and formalized the pen-and-paper proof by Gao & Rexford on the convergence of BGP, revealing necessary extensions to Gao & Rexford's original configuration guidelines; 2) we have built the Bagpipe tool which automatically checks that BGP configurations adhere to given policy specifications, revealing 19 apparent errors in three ASes with over 240,000 lines of BGP configuration; and 3) we have tested the BGP simulator C-BGP, revealing one bug.

1. Extending and Formalizing Gao & Rexford Gao & Rexford [6] proposed a set of guidelines for BGP router configuration, and they proved Internet-wide route convergence if these guidelines are implemented by every AS on the Internet.

The pen-and-paper proof by Gao & Rexford makes various simplifying assumptions about the BGP protocol. For example, routers have access to all the routes received by other routers within the same AS, routes are not transferred over a network but are instantly accessible whenever a router is activated, and route announcements cannot be withdrawn.

We used our semantics to extend and formalize Gao & Rexford's informal proof. Because our semantics of RFC 4271 eliminates the aforementioned simplifying assumptions, the proof requires additional insights. For example, because our semantics models both intra-domain and inter-domain routing, we have to prove intra-domain convergence of each AS, which requires an extension to Gao & Rexford's original guidelines.

2. Building Bagpipe Bagpipe[1] provides a declarative domain-specific language that enables BGP administrators to express control-plane policy specifications, such as "*an AS's routers will never accept routes for invalid IP addresses*", "*an AS's routers will always forward certain routes to other ASes*", and "*an AS's routers will always prefer routes from customers over routes from providers*". Given a specification expressed in this language, Bagpipe invokes an SMT solver to automatically verify that an AS's router configurations satisfy the given specification. Using our semantics, we formally verified that Bagpipe is sound, i.e. it will never falsely claim that an AS correctly

[1]Bagpipe is open-source, see `http://bagpipe.uwplse.org/`.

implements a specification.

Bagpipe's domain-specific language is rich enough to express specifications inferred from real AS configurations, express specifications found in the literature (such as the Gao-Rexford guidelines [6] and prefix-based filtering [12]), and express specifications for 10 configuration scenarios from the Juniper TechLibrary [9, 1]. The Bagpipe verifier works out-of-the-box for existing Juniper and Cisco router configurations, and the above specifications. To evaluate Bagpipe's efficiency, we applied it to three ASes with over 240,000 lines of Cisco and Juniper BGP configuration. Bagpipe found 19 apparent errors without issuing any false positives.

3. Testing C-BGP Using our semantics, we performed randomized differential testing [5] against C-BGP [13], a popular open-source BGP simulator. To this end, we developed a test harness which generates a random BGP network (including topology, router configurations, and initial routes) and then passes it to C-BGP. C-BGP runs a simulation of the BGP network, leading to a trace that captures all the route announcements exchanged by the routers in the BGP network, and the routes installed in each router's routing information bases. The test harness then checks that this trace is permitted under our semantics.

We ran this test harness over 100,000 times on randomly generated BGP networks. Some tests revealed that C-BGP occasionally sends announcements even when the routes they are advertising have not changed. This is not permitted by Section 9.2 of the BGP specification, and it is therefore rejected by our semantics. We reported this bug, the C-BGP maintainer acknowledged it, and we fixed it.

Conclusion We have defined a formal, mechanized semantics for BGP in Coq. We used it to extend and formalize BGP configuration guidelines, to build and verify a BGP checker, and to test a BGP simulator. These activities provide evidence that our semantics is correct and is useful for the development of reliable tools and guidelines that help BGP administrators avoid router misconfiguration.

References

[1] BGP feature guide for the OCX series (2015)

[2] Brown, M.: Pakistan hijacks YouTube. `http://research.dyn.com/2008/02/pakistan-hijacks-youtube-1/` (2008)

[3] Chiesa, M., Cittadini, L., Vanbever, L., Vissicchio, S., Di Battista, G.: Using routers to build logic circuits: How powerful is BGP? In: ICNP (2013)

[4] Cowie, J.: China's 18-minute mystery. `http://research.dyn.com/2010/11/chinas-18-minute-mystery/` (2010)

[5] Evans, R.B., Savoia, A.: Differential testing: A new approach to change detection. In: ESEC-FSE (2007)

[6] Gao, L., Rexford, J.: Stable Internet routing without global coordination. In: SIGMETRICS (2000)

[7] Goldberg, S.: Why is it taking so long to secure Internet routing? Queue (2014)

[8] Internet2 configurations. `http://vn.grnoc.iu.edu/Internet2/configs/configs.html`

[9] Junos OS: Routing policies, firewall filters, and traffic policers feature guide for routing devices (2016)

[10] Madory, D.: Chinese routing errors redirect Russian traffic. `http://research.dyn.com/2014/11/chinese-routing-errors-redirect-russian-traffic/` (2014)

[11] McConnell, D.: Chinese company 'hijacked' U.S. web traffic. `http://www.cnn.com/2010/US/11/17/websites.chinese.servers/` (2010)

[12] Meyer, D., Schmitz, J., Alaettinoglu, C.: Application of routing policy specification language (RPSL) on the Internet (1997)

[13] Quoitin, B., Uhlig, S.: Modeling the routing of an autonomous system with C-BGP. IEEE Network (2005)

[14] Rekhter, Y., Li, T., Hares, S.: A Border Gateway Protocol 4 (BGP-4). RFC 4271, Network Working Group (2006)

[15] Slane, D.: 2010 report to Congress of the U.S.–China Economic and Security Review Commission (2010)

[16] Turner, D., Levchenko, K., Snoeren, A.C., Savage, S.: California fault lines: Understanding the causes and impact of network failures. In: SIGCOMM (2010)

[17] Voellmy, A.R.: Proof of an interdomain policy: a load-balancing multi-homed network. In: SafeConfig (2009)