

FSR: Formal Analysis and Implementation Toolkit for Safe Inter-domain Routing

Yiqing Ren* Wenchao Zhou* Anduo Wang* Limin Jia†
Alexander J.T. Gurney* Boon Thau Loo* Jennifer Rexford‡
*University of Pennsylvania †Carnegie-Mellon University ‡Princeton University
{yiqingr, wenchaoz, anduo, agurney, boonloo}@cis.upenn.edu,
liminjia@cmu.edu, jrex@cs.princeton.edu

ABSTRACT

We present the demonstration of a comprehensive toolkit for analyzing and implementing routing policies, ranging from high-level guidelines to specific router configurations. Our Formally Safe Routing (*FSR*) toolkit performs all of these functions from the same algebraic representation of routing policy. We show that routing algebra has a very natural translation to both *integer constraints* (to perform safety analysis using SMT solvers) and *declarative programs* (to generate distributed implementations). Our demonstration with realistic topologies and policies shows how *FSR* can detect problems in an AS’s iBGP configuration, prove sufficient conditions for BGP safety, and empirically evaluate convergence time.

Categories and Subject Descriptors

C.2.1 [Computer-Communication Networks]: Network Architecture and Design

General Terms

Design, Languages, Experimentation

1. OVERVIEW

The Internet’s global routing system does not necessarily converge, depending on how individual networks configure their Border Gateway Protocol (BGP) policies. Since protocol oscillations cause serious performance disruptions and router overhead, researchers devote significant attention to BGP stability (or “safety”).

To aid the design, analysis, and evaluation of safe interdomain routing, we propose the *Formally Safe Routing (FSR)* analysis and implementation toolkit. *FSR* serves two important communities. For researchers, *FSR* automates important parts of the design process and provides a common framework for describing, evaluating, and comparing new safety guidelines. For network operators, *FSR* automates the analysis of internal router (iBGP) and border gateway (eBGP) configurations for safety violations. For both communities, *FSR* automatically generates realistic protocol implementations to evaluate real network configurations (e.g., to study convergence time) prior to actual deployment.

FSR bridges analysis and implementation, by combining *routing algebras* [3, 7] with recent advances in *declarative networking* [4] to produce provably-correct implementations of safe interdomain routing. Given policy configurations as input, *FSR* produces an analysis of safety properties and a distributed protocol implementation, as shown in Figure 1. In this section, we briefly summarize its main underlying technologies. Our technical report [8] provides details of the toolkit, including several examples and use cases.

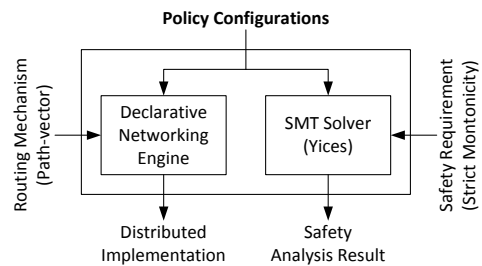


Figure 1: *FSR* Architecture.

Policy Configuration as Routing Algebra. *FSR* uses routing algebra [3, 7] to allow researchers and network operators to express policy configurations in an abstract algebraic form. A routing algebra is an abstract structure that describes how network nodes calculate routes, and the preference for one route over another. *FSR* uses our extensions to routing algebra for distinguishing import and export filtering policies, to enable automated translation from a policy configuration to a distributed protocol implementation. We support a wide range of policy configurations, ranging from high-level *guidelines* (e.g., the Gao-Rexford guideline [1]) to specific network configurations expressed as an instance of the Stable Paths Problem (SPP) [2].

Safety Analysis. Given any algebra, *FSR* fully automates the process of safety analysis, relieving users from the manual and error-prone process of proving safety for each new guideline or router configuration. The key insight is that the safety analysis can be translated automatically into integer constraints checkable by a standard SMT (Satisfiability Modulo Theories) solver. In a nutshell, a SMT solver determines whether a set of constraints (i.e., first-order logic formulas) are satisfiable with respect to a given background theory (e.g., integer theory).

Given a policy configuration written in routing algebra, *FSR* automatically generates integer constraints for safety analysis recognizable by the Yices SMT solver [9]. The

solver determines whether it is possible to jointly satisfy the policy configuration and the safety requirement of “strict monotonicity” (the rightmost input in Figure 1, drawn from previous work [7] on sufficient conditions for safety). If all constraints can be satisfied, the routing system is provably safe; otherwise, the solver outputs the minimal subset of the constraints that are not satisfiable to aid in identifying the problem and fine-tuning the configuration.

Provably Safe Implementations. To enable an evaluation of protocol dynamics and convergence time, *FSR* uses our extended routing algebra to automatically generate a distributed routing-protocol implementation that matches the policy configuration—avoiding the time-consuming and error-prone task of manually creating an implementation. Given the policy configuration and a formal description of the path-vector mechanism (the leftmost input in Figure 1), *FSR* generates a correct translation to a *Network Datalog* (*NDlog*) specification, which is then executed using the RapidNet declarative networking engine [5]. In practice, *FSR*’s safe implementation can be used as an emulation platform for studying BGP performance. (By changing the left input in Figure 1, researchers can also experiment with alternative routing mechanisms, as in Scenario Set D below.)

Our choice of *NDlog* is motivated by the following. First, the declarative features of *NDlog* allow for straightforward translation from the algebra to *NDlog* programs. Second, *NDlog* results in compact specifications that have orders of magnitude less code than imperative implementations. This makes possible a clean and concise proof (via logical inductions) of the correctness of the generated *NDlog* programs with regard to the algebra. The compact specifications also make it easy to incorporate alternative routing mechanisms to the basic path-vector protocol. Third, *NDlog*’s roots in logic and Datalog makes it amenable to the use of theorem provers to verify correctness properties. Finally, prior work [4] has shown that these declarative networks perform efficiently relative to imperative implementations.

2. DEMONSTRATION PLAN

FSR provides a graphical user-interface for users to specify policy configurations using algebraic specifications, which are compiled into *NDlog* programs and executed on RapidNet. This engine allows for a *simulation mode* in ns-3 [6], enabling comprehensive examination under various network topologies and conditions, as well as a *deployment mode* where different hosts in a testbed environment execute the deployed system over a real network.

In our demonstration, network traces obtained from actual *NDlog* execution runs are directed to the RapidNet visualizer. The visualizer displays the network topology, routing state, alongside actual performance statistics such as bandwidth utilization. Our demonstration showcases the following four scenario sets:

Scenario Set A: Provably Safe Guidelines. Our first case study presents scenarios where a researcher empirically evaluates policy guidelines using the distributed *NDlog* implementation automatically generated from the algebraic specifications. The researcher uses the *FSR* tool to analyze policy guidelines for safety properties using Yices. In addition, the researcher can use the declarative *NDlog* implementation generated by *FSR* to study the actual behavior of the protocol when executed under the guideline constraints,

for instance, to measure the convergence time with respect to the depth of the AS hierarchy.

Scenario Set B: Pinpoint iBGP Configuration Errors. We also emulate scenarios where a network operator uses our *FSR* toolkit to study the safety properties of existing iBGP network configurations. We use the intradomain topology from the Rocketfuel dataset as a basis. To experiment with *FSR*’s ability to detect configuration errors in large network instances, we embed gadgets that are known to cause oscillation in an iBGP setting. *FSR* detects the errors in both its analysis and actual execution runs. In our demonstration, we fix these configuration errors pointed out by Yices, and demonstrate that the resulting iBGP configuration is safe, both in analysis and implementation.

Scenario Set C: eBGP Gadget Analysis. *FSR*’s applicability extends beyond high-level guidelines and iBGP configurations. Our third scenario is to use *FSR* to analyze well-known eBGP gadgets, such as GOODGADGET, BADGADGET and DISAGREE [2] when embedded into large network instances. We execute these embedded gadgets using the automatically-generated *NDlog* implementation, to visually study their behavior in actual execution.

Scenario Set D: Alternative Routing Mechanisms. In our final scenario set, we demonstrate the flexibility of the *FSR* toolkit to support alternative routing mechanisms beyond the basic path-vector protocol, for instance, the *Hybrid Link-State and Path-Vector Protocol* (HLP).

Acknowledgments. This research was supported by NSF grants IIS-0812270, CCF-0820208, CNS-0830949, CNS-0845552, CNS-1040672, AFOSR Grant No: FA9550-08-1-0352, ONR Grant No: N00014-09-1-0770, and a gift from Cisco Systems.

3. REFERENCES

- [1] GAO, L., AND REXFORD, J. Stable Internet routing without global coordination. In *ACM SIGMETRICS* (2000).
- [2] GRIFFIN, T. G., SHEPHERD, F. B., AND WILFONG, G. The stable paths problem and interdomain routing. *IEEE/ACM Trans. on Networking* 10 (2002).
- [3] GRIFFIN, T. G., AND SOBRINHO, J. L. Metarouting. In *ACM SIGCOMM* (2005).
- [4] LOO, B. T., CONDIE, T., GAROFALAKIS, M., GAY, D. E., HELLERSTEIN, J. M., MANIATIS, P., RAMAKRISHNAN, R., ROSCOE, T., AND STOICA, I. Declarative networking. In *Communications of the ACM* 52 (2009).
- [5] MUTHUKUMAR, S. C., LI, X., LIU, C., KOPENA, J. B., OPREA, M., AND LOO, B. T. Declarative toolkit for rapid network protocol simulation and experimentation. In *ACM SIGCOMM (demo)* (2009).
- [6] NETWORK SIMULATOR 3. <http://www.nsnam.org/>.
- [7] SOBRINHO, J. An algebraic theory of dynamic network routing. *IEEE/ACM Trans. on Networking* 13 (2005).
- [8] WANG, A., JIA, L., ZHOU, W., REN, Y., LOO, B. T., REXFORD, J., NIGAM, V., SCEDROV, A., AND TALCOTT, C. FSR: Formal analysis and implementation toolkit for safe inter-domain routing. University of Pennsylvania Tech. Report MS-CIS-11-10 (2011), http://repository.upenn.edu/cis_reports/954/.
- [9] YICES. <http://yices.csl.sri.com/>.