Declarative Toolkit for Rapid Network Protocol Simulation and Experimentation

Shivkumar C. Muthukumar* Xiaozhou Li* Changbin Liu* Joseph B. Kopena†
Mihai Oprea* Boon Thau Loo*
*University of Pennsylvania †Drexel University
{mshivk, xiaozhou, changbl, mihaio, boonloo}@seas.upenn.edu, tjkopena@cs.drexel.edu

ABSTRACT

We propose the demonstration of the *RapidNet* toolkit for rapid network protocol simulation, implementation and experimentation. *RapidNet* utilizes *declarative networking*, a declarative, database-inspired extensible infrastructure that uses query languages to specify behavior. *RapidNet* integrates a declarative networking engine with the emerging ns-3 network simulator. Our proposed demonstration will showcase two recent use cases: declarative mobile ad-hoc network (MANET) routing and network composition.

Categories and Subject Descriptors

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

General Terms

Design, Languages, Experimentation

1. INTRODUCTION

RapidNet is a development toolkit that enables rapid simulation, implementation, and experimentation of network protocols. *RapidNet* integrates a declarative networking [8, 7] engine with the emerging ns-3 [10, 4] network simulator, which is intended as an eventual replacement for the ns-2 simulator. Network protocols are specified using declarative specifications, which are then compiled into ns-3 code for simulation and analysis. The same declarative specifications can also be used as actual implementations using the P2 declarative networking system [1] or the ns-3 network emulator. We will demonstrate the use of *RapidNet* in recent research projects that use declarative networking: (1) declarative mobile ad-hoc network (MANET) routing [6] and network composition [9].

The high level goal of *declarative networks* is to build extensible architectures that achieve a good balance of flexibility, performance and safety. Declarative networks are specified using *Network Datalog* (*NDlog*), which is a distributed recursive query language for querying networks. Declarative queries such as *NDlog* are a natural and compact way to implement a variety of routing protocols and overlay networks. For example, traditional routing protocols such as the path vector and distance-vector protocols can be expressed in a few lines of code [8], and the Chord distributed hash table (DHT) in 47 lines of code [7]. When compiled and executed, these declarative networks perform efficiently relative to imperative implementations.

The long term goal of *RapidNet* is to provide a platform for rapid prototyping, synthesis, and deployment of new network protocols that can be provably verified prior to deployment. In addition to being a valuable tool for rapid network prototyping and analysis, *RapidNet* can potentially be used as a basis of an educational software package that integrates the declarative platform with the ns-3 simulator, enabling students to learn about network protocols via higher level declarative abstractions.

2. OVERVIEW

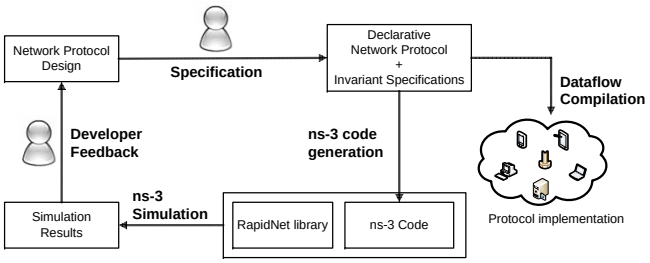

Figure 1: Overview of *RapidNet*

Figure 1 provides an overview of *RapidNet*'s basic approach towards unifying specifications, simulation, and implementation within a common declarative framework. In the initial design phase of *RapidNet*, a network protocol design is used as the basis for specifying the network protocol using the *NDlog* declarative networking language. High-level invariant properties of the protocol can also be expressed in *NDlog* as *distributed triggers* which raise event alarms when invariants are violated.

In the *simulation mode*, the *RapidNet* compilation process generates ns-3 code from the *NDlog* protocol specifications and invariants. The generated code either runs as an ns-3 application, or replaces routing protocol implementations at the network layer. The generated code implements dataflows (execution plans) with a similar execution model with the Click modular router [5], which consists of elements that are connected together to implement a variety of network and flow control components. In addition, those elements include database operators (such as joins, aggregation, selection, and projection) that are directly generated from the declarative networking rules. Messages flow among dataflows executed at different nodes, resulting in updates to local tables. The local tables store the state of intermediate

and computed query results which include the network state of various network protocols. In the *implementation mode*, declarative networking specifications are directly executed and deployed either by using the P2 declarative networking system [1] or the ns-3 network emulator.

Since declarative networks share common functionalities such as the network stack, multiplexing tuple messages entering and leaving the dataflow, and database functionalities, all these utilities are defined in a shared *RapidNet* library. This enables one to simplify the compilation process to only the relevant database operations to implement the distributed dataflows for the corresponding declarative network specification. This also enables one to easily incorporate *multi-query optimizations* to share computations across declarative networks in future.

3. DETAILS OF DEMONSTRATION

Our demonstration takes as input declarative network specifications which are automatically compiled to ns-3 code for execution in the ns-3 simulation and emulation modes. Network traces are directed to a ns-3 visualizer [11] that will display the actual movement of nodes during the simulation, side-by-side actual performance statistics of the protocol obtained from the ns-3 network statistics package. To illustrate, Figure 2 shows an example execution of the current version of our demonstration. A declarative path-vector protocol is automatically compiled into ns-3 code, and runs within an ns-3 scenario where 90 nodes within an arena communicate via 802.11b ad-hoc mode and move with Brownian motion model.

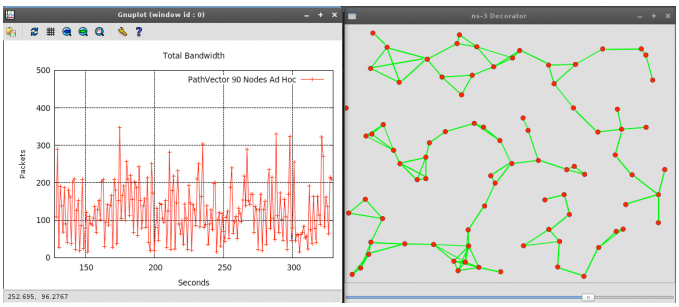

Figure 2: Screenshot of real-time network statistics (left) and the network visualizer (right).

Based on the above setup, we will demonstrate the following recent use cases of declarative networking:

Declarative MANET routing: Our first use case involves the declarative MANET protocol implementations, ranging from link-state routing (LS), hazy-sighted link-state routing(HSLS) [12], optimized link-state routing (OLSR) [3], dynamic source routing (DSR), and summary-vector based epidemic routing. Table 1 summarizes those protocols, by categorizing them as *proactive*, *reactive*, and *epidemic*, as well as respective number of rules. These MANET protocols will be evaluated using different mobility models (e.g. random waypoint, Brownian motion, hierarchical mobility, etc.) supported by ns-3. This first use case provides us with a wide-range of examples to demonstrate the use of *RapidNet* for effective prototyping, deployment, and comparisons across a variety of MANET protocols. Beyond comparisons across protocols, the declarative framework enables the ability to rapidly explore a wide range of deployment and implementation parameters necessary for tuning the performance of MANET routing protocols.

Category	Protocol	Rules
Reactive	Dynamic source routing	11
Proactive	Traditional link state	15
	Optimized link state routing	34
	Hazy sighted link state	18
Epidemic	Summary-vector based epidemic	17

Table 1: Declarative MANET Protocols.

Network composition: Our second use case is on the MOSAIC [9] network composition platform, which provides a declarative platform that enables distinct parts or elements of existing networks to be combined to create a new network with new functionalities. Each component declarative network is specified as a *composable view*, essentially a group of rules represented with a single predicate, and then composed (via *bridging* and *layering*) with other component networks achieved via additional *NDlog* rules. Example compositions include layering a declarative Chord DHT implementation over a resilient overlay network (RON) [2] for robustness, and composing an indirection overlay [13] with RON for robust mobility.

4. ACKNOWLEDGMENTS

This work is based on work supported in part by NSF grants CNS-0721845, CCF-0820208, and CNS-0845552.

5. REFERENCES

[1] P2: Declarative Networking System. http://p2.cs.berkeley.edu.
[2] D. Anderson, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient Overlay Networks. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2001.
[3] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr). In *RFC 3626 (Experimental)*, 2003.
[4] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. B. Kopena. Network simulations with the ns-3 simulator. In *SIGCOMM Demonstration*, 2008.
[5] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click Modular Router. *ACM Transactions on Computer Systems*, 18(3):263–297, 2000.
[6] C. Liu, Y. Mao, M. Oprea, P. Basu, and B. T. Loo. A declarative perspective on adaptive manet routing. In *ACM SIGCOMM Workshop on Programmable Routers for Extensible Services of TOmorrow*, 2008.
[7] B. T. Loo, T. Condie, J. M. Hellerstein, P. Maniatis, T. Roscoe, and I. Stoica. Implementing Declarative Overlays. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2005.
[8] B. T. Loo, J. M. Hellerstein, I. Stoica, and R. Ramakrishnan. Declarative Routing: Extensible Routing with Declarative Queries. In *Proceedings of ACM SIGCOMM Conference on Data Communication*, 2005.
[9] Y. Mao, B. T. Loo, Z. Ives, and J. M. Smith. MOSAIC: Unified Declarative Platform for Dynamic Overlay Composition. In *4th Conference on emerging Networking EXperiments and Technologies (ACM CoNEXT)*, 2008.
[10] Network Simulator 3. http://www.nsnam.org/.
[11] ns 3 visualizer. http://code.nsnam.org/tjkopena/ns-3-decorator3/.
[12] C. Santivanez, R. Ramanathan, and I. Stavrakakis. Making link-state routing scale for ad hoc networks. In *ACM MobiHoc '01*, Long Beach, CA, 2001.
[13] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *Proceedings of ACM SIGCOMM Conference on Data Communication*, 2002.