ENSC-891: Directed Studies

ns-BGP integration with ns-2.33

Summer 2008

Will Hrudey

whrudey@sfu.ca

1 Abstract

The border gateway protocol (BGP) is an inter-Autonomous System (AS) routing protocol utilized as the core routing protocol in the Internet today. It was formally proposed in Request For Comments (RFC) 1771 by the network working group within the Internet Engineering Task Force (IETF). Primarily, BGP exchanges network reachability information with other BGP systems. Since BGP performance is affected by the dynamic nature of the Internet, ns-BGP was developed for the ns-2 network simulator in 2003 to facilitate realistic, flexible BGP routing experimentation [1].

In parallel with the ns-BGP development, academic and research communities continued to develop ns-2. Consequently, this led to an incompatible ns-BGP module with current versions of the simulator. Therefore, in an effort to aid further BGP research efforts, this project will integrate ns-BGP with the latest stable version of the simulator thereby benefitting from core ns-2 feature enhancements and maintenance updates over the past five years.

Table of Contents

List of Figures

List of Tables

2 Introduction

The border gateway protocol (BGP) [2,5,8] is used to exchange routing details between networks. It discovers multiple paths between networks using both internal and external BGP speaker nodes and then selects the optimal path to be configured in the routing table. This optimal path, which is influenced by corporate policies, then propagates to external BGP peers accordingly. Since BGP is a very robust and scalable routing protocol, it has become the *de facto* inter-domain routing standard with universities, corporate enterprise networks and Internet Service Providers (ISP).

Given the widespread adoption of BGP and the shear size of the Internet, BGP simulations provide more practical, flexible experiments than both theoretical and empirical studies [1]. Consequently, BGP support was developed for the widely recognized network simulation tool, ns-2 [3]. The BGP module, ns-BGP, was developed natively in ns-2.26 in 2003 [1]. This effort involved the port of BGP version 4 as well as the socket layer from SSFNet [9], a Java-based simulator.

However, since this effort took place in 2003, ns-2 has seen continued development by academic and research communities' thereby preventing transparent re-integration of ns-BGP into the latest ns-2 release. Consequently, in an effort to aid further BGP research efforts, this project will integrate ns-BGP with the latest stable version of the simulator that includes feature enhancements and maintenance updates over the past five years. Figure 1 illustrates the project scope.

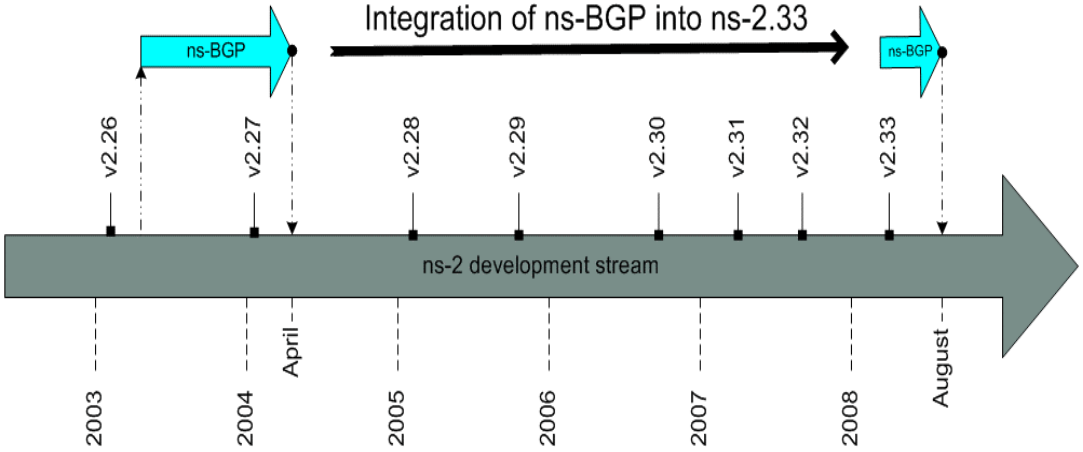

Figure 1. Project scope.

The project is organized into various phases that are described below:

⇨ *Technology fundamentals and review phase*

Given the low level integration required to achieve the stated project objective, significant effort throughout this project has been allocated to understanding the following technologies:

- ❑ C++ Object oriented programming language [6]
- ❑ Tcl/Tk [7]
- ❑ ns-2.33 network simulator [3,4]
- ❑ BGP-4 [2,5,8]

⇨ *Native ns-BGP development environment setup phase*

This phase encapsulates the efforts required to provision an equivalent development environment necessary to run ns-BGP 2.0 natively without modification in ns-2.26 or ns-2.27. ns-BGP was originally developed and tested in ns-2.26 and subsequently retested in ns-2.27. Tasks in this phase include the systematic derivation of a compatible environment that accommodates the compiler dependencies that ns-BGP was developed against. It includes the iterative installation and configuration of a compatible Linux operating system distribution, ns.2.26 / ns-2.27, and ns-BGP 2.0 until a working configuration is identified.

Ultimately, this environment will serve as the functional benchmark in the project validation stage to which the target, integrated ns-2.33 environment will be compared against. Specifically, the output of each ns-BGP test script in a ns-2.26/2.27 environment will be compared against the equivalent test script output in ns-2.33.

⇨ *Target ns-BGP integration environment setup phase.*

This phase encompasses the efforts required to provision the target ns-2.33 integration environment. Tasks in this phase include the installation and configuration of the Fedora Core 8 Linux distribution and ns-2.33.

⇨ *ns-BGP release analysis phase*

This phase involves the systematic inspection and analysis of the native ns-BGP 2.0 software release archive [1]. The objective of this phase is to understand the scope of the ns-BGP software implementation as well as the source file set and file distribution across an ns-2 build environment. Additionally, it identifies core ns-2 source file changes which will drive the code integration stage.

⇨ *Code integration phase*

This phase consists of two sub-phases. The first sub-phase manually propagates non-overlapping ns-BGP source files to the target ns-2.33

integration environment. The second sub-phase involves the detailed inspection and analysis of the code differences between overlapping core ns-2 source files between ns-2.26/2.27 and ns-2.33. The resulting code differences are then propagated to the equivalent source file in ns-2.33.

⇨ *Code compilation phase*

This phase is comprised of the systematic compilation of ns-BGP in the newly integrated ns-2.33 environment; the very nature of software integration necessitates full recompilation of the entire ns-2 build environment. This phase includes the inevitable iterative compile / fix / recompile sequence.

⇨ *Project validation phase*

This key phase encompasses the steps required to validate the ns-BGP integration into ns-2.33. Validation of the project will be achieved by way of two discrete phases. First, the successful recompilation of the target environment will validate the resulting integrated code syntax. Secondly, the individual comparison of the ns-BGP test script outputs between the native ns-2.26/2.27 and the target ns-2.33 environments will further validate the BGP functional equivalency between the two systems.

⇨ Project deliverables phase

This phase encompasses the tasks relating to the preparation of the project report, presentation slides, and project software release contents.

2.1 ns-2 Overview

The network simulator version 2, more commonly referred to as ns-2, is a widely recognized simulation tool in the academic and research communities. While it dates back to 1989 as the REAL network simulator, it was a DARPA VINT project in 1995 which was largely developed to study network protocol interactions and scalability [3]. Consequently, it is open source and freely distributed thereby nurturing and promoting continued development and maintenance. It is comprised of approximately 200,000 lines of code. Moreover, it is packaged with more than 200 test suites and examples; release target platforms include: FreeBSD, Linux, Solaris, Windows and MAC.

In order to address simulation complexity and processing time while providing a flexible configuration interface to accommodate ad-hoc, explorative simulation scenarios, ns-2 was implemented in both C++ and OTcl. Detailed protocol algorithms, packet generation and processing was written in C++ to exploit the power of object oriented software construction while leveraging off the execution speed of compiled logic. The configuration interface was written in OTcl which exploits the real-time interactive interface of an interpreted language.

As a flexible simulation tool, ns-2 supports both wired and wireless technologies; various routing algorithms, transport protocols (TCP, UDP, SCTP, RTP), traffic sources, queuing disciplines, and quality of service (QoS) are natively supported. Additionally, utilities including tracing, the visual network animator (nam), and topology / traffic generators are packaged in the release. Standard UNIX text parsing tools including sed, awk, and perl can be used to post-process ns-2 output files.

2.2 BGP Overview

The first version of BGP (BGP-1) was published in 1989 as RFC 1105. A second revision to BGP was released as BGP-2 in 1990 and described in RFC 1163. Almost a year and a half later, BGP-3 was released and described in RFC 1267. The current version, BGP-4, was released in 1994 and 1995 and described in RFC 1654 and 1771 respectively. Since then, several enhancements including multi-protocol extensions and MPLS/VPN support has been added to the BGP specification.

BGP is in itself a routing protocol which is primarily used to exchange network layer reachability information (NLRI) between autonomous systems (AS). AS's are collections of routers under a single administration; they can be a single network or multiple networks which use the same routing protocols called interior gateway protocols (IGP). IGP is used to route packets within an AS. The routing information protocol (RIP), open shortest path first (OSPF) and interior gateway routing protocol (IGRP) are common IGP protocols. As an exterior gateway protocol (EGP), BGP is used to route traffic between AS's and consequently uses the reliable transport services of TCP on port 179 for peer connections. Figure 2 illustrates the AS interconnections and their corresponding internal and external routing protocols.

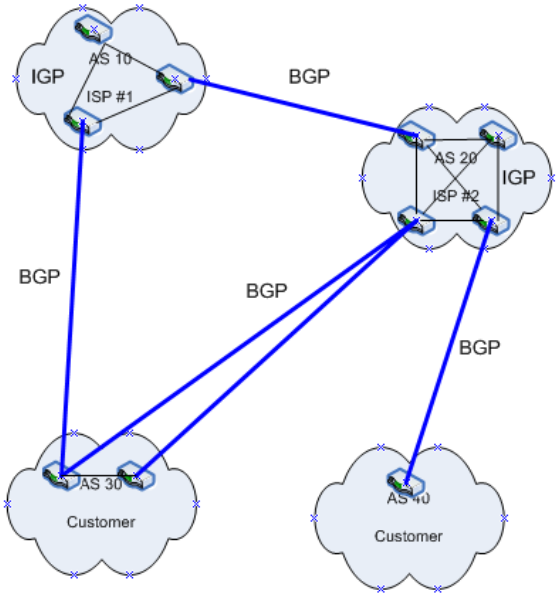

Figure 2. Network architecture routing protocols.

BGP connections are established between two BGP speakers which are essentially subsystems within a router that generate and process BGP specific messages. The message exchange between BGP speakers is driven by routing policies (announce and accept) which dictate which routes are used by a given router and which routes are transmitted to other routers. The connection between two BGP routers, known as BGP peering, involves the exchange of connection parameters, the entire routing table (initially), as well as incremental updates when the routing table changes.

This message information is communicated using the following message types:

- ❑ Open - first message sent by each BGP peer
- ❑ Keepalive - "ping" type message sent by each peer every 30 seconds
- ❑ Update - facilitate routing information to BGP peer
- ❑ Notification - sent by a peer when an error condition has occurred

The following message exchange presented in Figure 3 exemplifies a BGP peer connection.

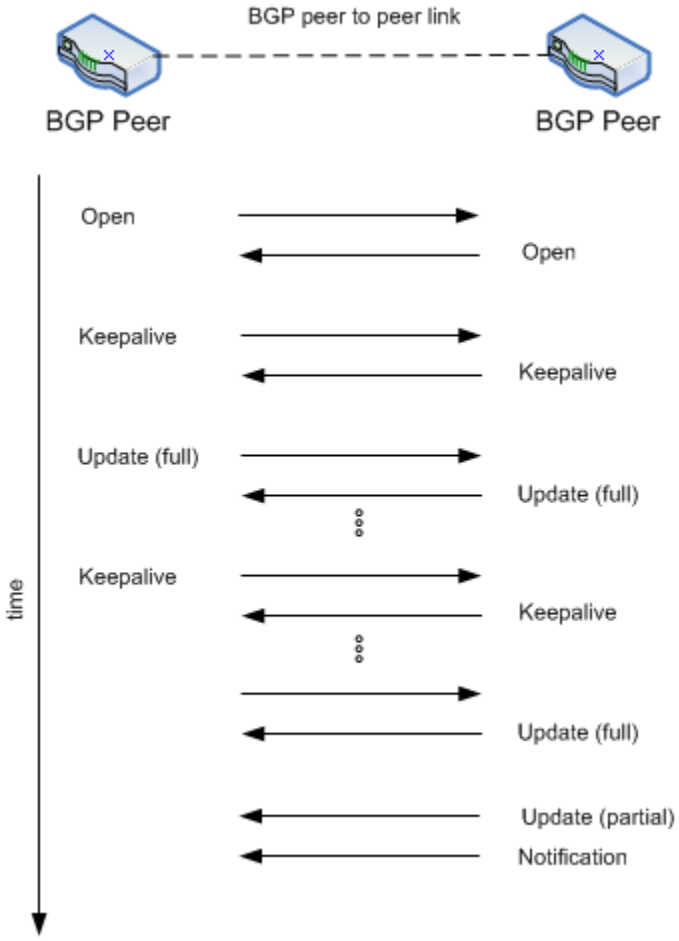

Figure 3. BGP peer message exchange.

The keepalive messages in Figure 3 are used to keep the connection alive. Full updates flow in either direction, while notifications are used to indicate fatal errors.

While BGP is an EGP, BGP is required within an AS to ensure reachability within a network before propagating information to other AS's. To make this distinction, interior BGP (iBGP) describes the peering between routers within an AS and exterior BGP (eBGP) describes the peering between routers from different AS's.

BGP polices control and modify routing information; these policies determine the conditions for redistributing routes from one protocol to another. "Accept" policies handle incoming route selection and "Announce" policies dictate the distribution of outgoing routes. These policies are configured into BGP rather than being apart of the protocol itself. Additionally, BGP updates can be filtered on route information, AS-path information, communities, AS-regular expression.

2.3 ns-BGP Overview

The ns-2 BGP module, formerly known as ns-BGP, was ported and adapted to ns-2 in 2003 [1]. This effort involved the port of the BGP and TcpSocket modules from SSFNet [9]. SSFNet is a Java-based network simulator which is comprised of a simulation engine and a configuration language known as the Domain Modeling Language (DML). Since SSFNet was implemented using object oriented technology, its BGP module was a natural candidate for ns-2 integration. Additionally, IPv4 addressing and packet forwarding was also incorporated into ns-2 to accommodate the SSFNet BGP dependencies.

Within ns-2, unicast routing is achieved using forwarding and control planes where the forwarding plane classifies and forwards packets to their destinations nodes using classifier and routing modules while the control plane provisions the route creation, computation, routing algorithms, and management of the routing tables. Figure 4 illustrates the ns-BGP unicast structure which is based the native ns-2 unicast architecture.

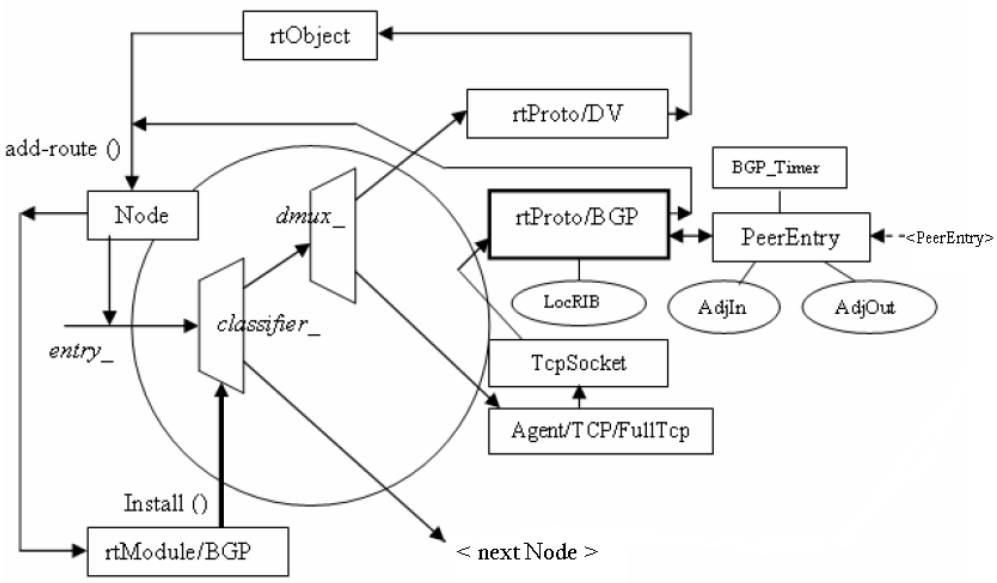

Figure 4. Unicast structure of ns-BGP.

As indicated in Figure 4, classifier modules such as address and port classifiers either send incoming packets to its respective agent or to an outgoing link. These modules are managed by the routing module. The control plane is comprised of the following components: route logic, route object, route peer, routing protocol. The central routing table is maintained by the routing logic while routing objects are used in dynamic routing simulations. Routing peer objects encapsulate the routing protocol by capturing and retaining attributes of each route advertised. Lastly, specific routing algorithms are implemented by the routing protocol components.

Moreover, since an ns-BGP node is derived from a ns-2 unicast node, Figure 4 denotes the inclusion of the BGP specific modules and TcpSocket modules. The rtModule/BGP module manages the IPv4Classifer object while the new routing protocol, rtProto/BGP, relies on the TcpSocket modules for packet transmission. Peer entry objects, which are used to establish and exchange BGP messages and close peer sessions, are allocated for each BGP peer.

Four key classes used in the BGP implementation are: TcpSocket, IPv4Classifer, rtModule/BGP, and rtProto/BGP. The TcpSocket class implements a UNIX-like socket application programming interface (API) which includes standard calls such as bind(), connect(), listen(), close(), read(), and write(). Additionally, it also provisions data queuing and callback functions. It is added to the modified FullTcpAgent which encapsulates the TCP services into a socket interface. The IPv4Classifier class is used to classify incoming packets by observing the destination address and forward it accordingly based on the routing table. It is implemented using a dual class whereby the class specification is defined in both C++ and OTcl. The rtModule/BGP provisions a registration interface to which active route modules must register with when a node is created. This module is

written exclusively in Tcl and it replaces the existing basic routing module. Finally the rtProto/BGP module implements the BGP-4 protocol as a dual class in both C++ and OTcl. Protocol operations include peer connection establishment, path selection, routing table updates, finite state machine (FSM) management.

Overall, ns-BGP is RFC 1771 compliant but does not support multi-protocol extensions. Optional features included in the release are:

- Multiple Exit Discriminator (MED)
- Aggregator
- Community
- Originator ID
- Cluster list path attributes
- Route reflections

Experimental features included in the release are:

- Sender-side loop detection
- Withdrawal rate limiting
- Unjittered minimum route advertisement interval timer
- Per-peer and per-destination rate limiting

3 Technical Environment

The initial computing platform used for this project was as follows:

- Toshiba Tecra S2 laptop
  - Intel Pentium M processor 1.86 GHz
  - 1 GB RAM
  - 80 GB HDD
  - Microsoft Windows XP Service Pack (SP) 3

However, approximately 75% through the project, the hardware was replaced by the following:

- Dell D630 laptop
  - Intel Duo core T7250 2.0 GHz processor
  - 4 GB RAM
  - 110 GB HDD
  - Microsoft Vista Business Edition SP1

The following software components were utilized in this project:

- VMware Server 1.0.6
- Cygwin
- Fedora Core 2
- Fedora Core 4
- Fedora Core 8

- ns-2.26
- ns-2.27
- ns-2.33
- ns-BGP 2.0 (for ns-2.26 / ns-2.27)

The following configurations were derived over the course of this project. Initially an environment using Cygwin was targeted given the convenience of switching between different ns-2 releases within a single OS instance. However, over time, it became evident that a given ns-2 release has significant dependencies on compiler version it was developed with. Consequently, a native Linux environment capable of running ns-2.26/2.27 became essential.

The Fedora Core (FC) distribution (formerly branded as Red Hat) was selected given its widespread adoption in the Linux community. FC version 8 was the current distribution release at the time the project commenced; this distribution incorporated a Linux 2.6.23.1-42 kernel. Additionally, ns-2.33 was the current release thereby defining the target integration environment of FC8 / ns-2.33 for this project.

In order to validate the integration of ns-BGP 2.0 into the target environment, a test environment was derived to recreate the ns-BGP development environment to run the ns-BGP test scripts natively. The output of these test environment scripts would then serve as the output reference to which the target environment would be compared against. This effort consequently resulted in the systematic downgrading of FC releases until a release that resulted in error-free compilation of ns-2.27 was achieved. Table 1 details the configurations.

System Configurations										
Derived configurations	Software									
	Vista Business Edition	VMWare Server 1.06	WinXP SP3 / Cygwin	Fedora Core 2	Fedora Core 4	Fedora Core 8	ns-2.26	ns-2.27	ns-2.33	ns-BGP 2.0 (original release)
Test-1	▪	▪	▪				▫	▫	▪	▫
Test-2 (target integration environment)	▪	▪				▪	▫	▫	▪	▫
Test-3	▪	▪			▪		▫	▫		
Test-4 (native ns-BGP development environment)	▪	▪		▪			▪	■		▪

Table 1. System configurations.

The ▪ symbol denotes the successful inclusion of a given software component while the ▫ denotes the unsuccessful inclusion of a given component. Ultimately, test-2 represents the target integration environment, while test-4 represents the native ns-BGP test environment. Since ns-BGP was initially developed for ns-2.26 and subsequently re-released for ns-2.27, this project attempted to successfully compile both versions, with a bias to ns-2.27 given that it is more recent than ns-2.26. Using VMware Server, these Linux environments are running in virtual machines (VM). Figure 5 details an ns-2.27 session running in the FC2 VM.

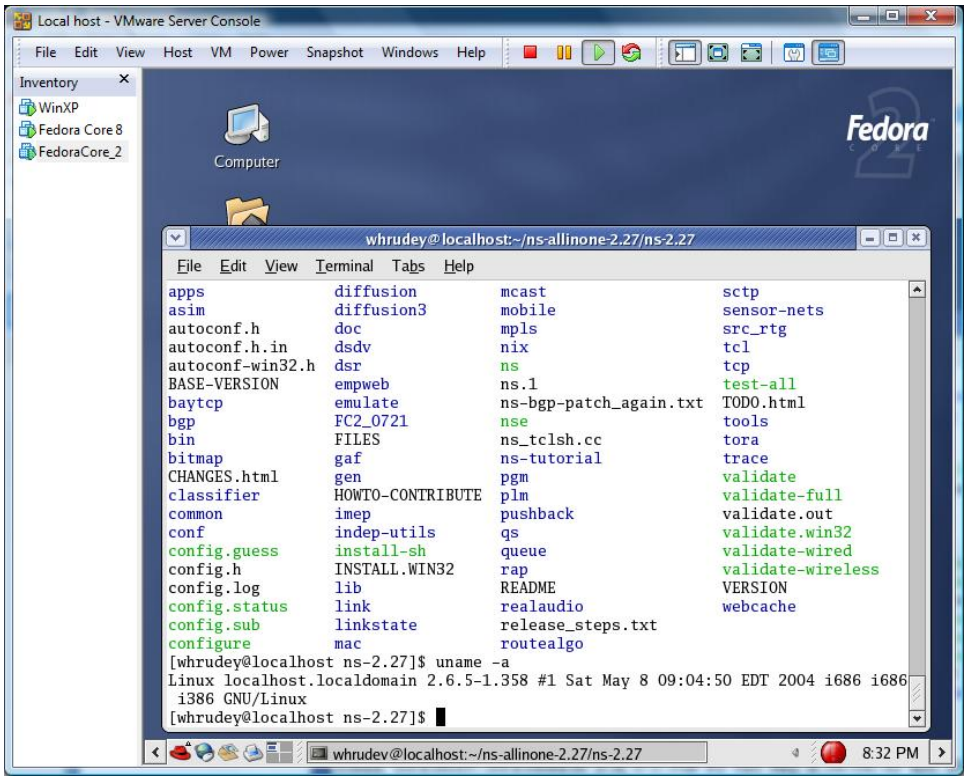

Figure 5. Fedora Core 2 virtual machine running ns-2.27.

Additionally, Figure 6 details an ns-2.33 session running in the FC8 VM

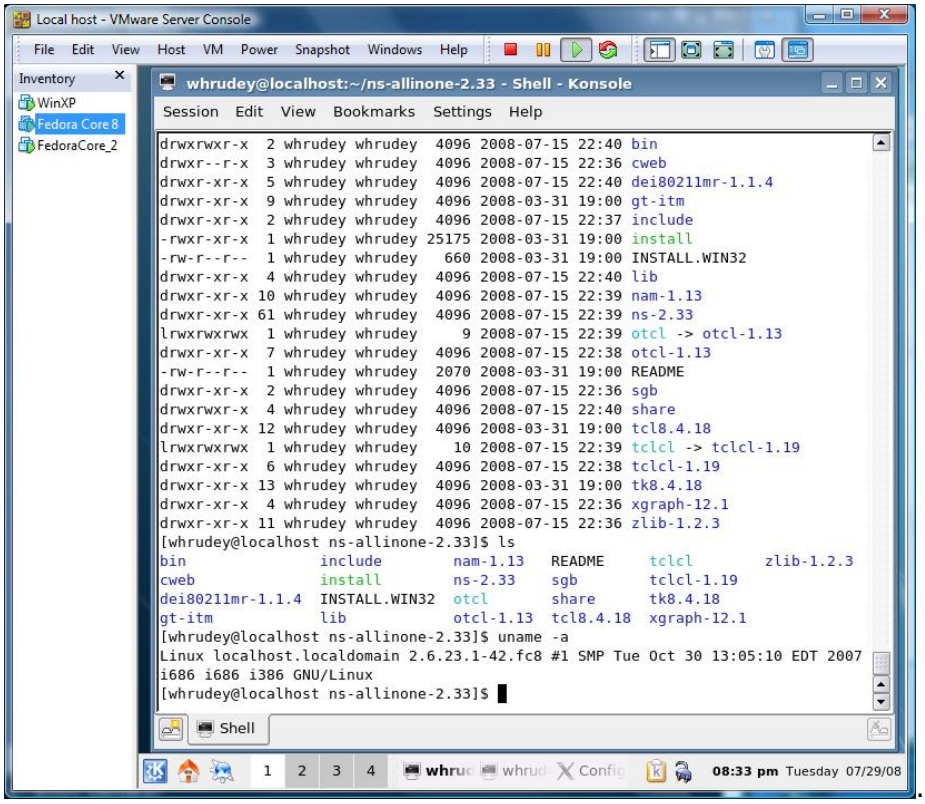

Figure 6. Fedora Core 8 virtual machine running ns-2.33.

4 ns-BGP Analysis

The ns-BGP 2.0 release is provisioned as a software “archive” comprised of a series of source and header files distributed across a series of directories. See Appendix 11.3 for a complete file listing. As illustrated in Figure 7 the file set is organized into a series of directories and subdirectories.

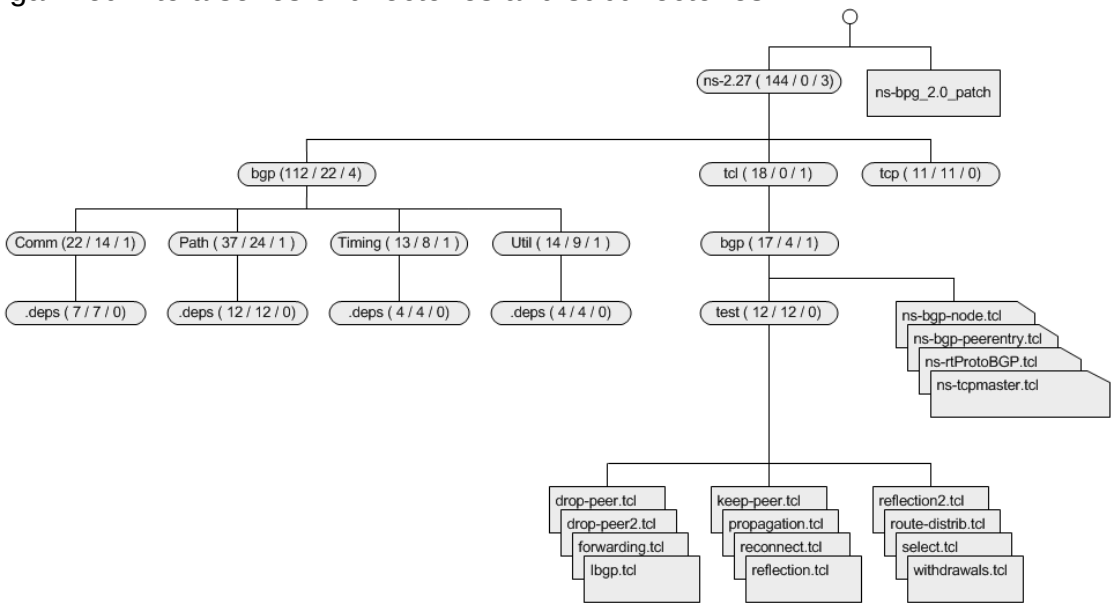

Figure 7. ns-BGP source file hierarchical structure.

The notation used in Figure 7 is as follows:

- ❑ rounded rectangles represent directories
- ❑ integers in parenthesis (x / y / z)
  - ⇒ x – all files and subdirectories in current directory and below
  - ⇒ y – files in current directory only
  - ⇒ z – subdirectories in current directory only
- ❑ angled rectangles represent core ns-BGP Tcl scripts
- ❑ rectangles ns-BGP test scripts

Not all the files could be graphically displayed in Figure 7 given the total number of files. Overall, there were 104 source files categorized as: 41 C++ files, 46 header files, 16 Tcl files, and one patch file.

The release package is intended to be extracted into the ns-2 top level directory, thereby performing the following tasks [1]:

- ❑ Creates a patch file in the top level directory
- ❑ Creates a bgp subdirectory with source files in the ns-2 subdirectory
- ❑ Adds new files for TcpSocket to the ns-2/tcp subdirectory
- ❑ Creates a bgp subdirectory under the tcl directory with all the test scripts

Then the patch file is processed to apply the appropriate patches to various core ns-2 source files. Once the patches have been applied, ns-2 must be recompiled.

5 Integration

The objective of the integration phase is to migrate the ns-BGP module, which was designed and developed for ns-2.27, into the current version of the simulator: ns-2.33. Additionally, all subsequent ns-2 enhancements and maintenance updates that have occurred since ns-BGP was released must be retained. In order to achieve the stated objectives, a two-stage approach was devised, which was comprised of a code integration stage and a compilation stage.

The code integration stage involved the careful propagation of logic contained in new BGP specific files as well as logic changes to existing core ns-2 files detailed in the patch file. As indicated in the ns-BGP analysis section, there were 145 files in the release, of which 104 files were C++ code files, C++ header files, and script files. The code migration strategy was to migrate the source files and the Readme.txt file to their respective target ns-2.33 directory locations. The dependency files (.Po) were not required. The patch file contained edits to 16 core ns-2 files as indicated in the merge column of Table 2. In the table, a distinction is made between basic and moderate logic integration using x and **X** respectively.

The ns-BGP patch file provisioned with the release was no longer valid given the subsequent development between the initial ns-BGP ns-2 development version (ns-2.27) and the current ns-2 version (ns-2.33). As a result, the context code and code line numbers contained in the patch file, which were key to the correct insertion of BGP logic, was well out of date. Consequently, in an effort to ensure subsequent core ns-2 enhancements and bug fixes were retained as a part of the integration phase, the specific ns-BGP logic edits to an ns-2.27 core source file was identified and then migrated into the corresponding ns-2.33 target file.

Files	Changes	
	Merge	Compilation
ns-2.33/common/node.cc	o	
ns-2.33/common/node.h	o	
ns-2.33/common/packet.h	X	
ns-2.33/common/simulator.cc	X	
ns-2.33/Makefile.in	o	
ns-2.33/routing/route.cc	o	
ns-2.33/routing/rtmodule.cc	o	
ns-2.33/routing/rtmodule.h	o	
ns-2.33/tcl/lib/ns-default.tcl	o	
ns-2.33/tcl/lib/ns-lib.tcl	o	
ns-2.33/tcl/lib/ns-node.tcl	X	
ns-2.33/tcp/rq.cc	o	
ns-2.33/tcp/rq.h	o	
ns-2.33/tcp/scoreboard-rq.cc	o	
ns-2.33/tcp/tcp-full.cc	X	
ns-2.33/tcp/tcp-full.h	X	
ns-2.33/bgp/Util/ipaddress.cc		X
ns-2.33/bgp/Util/ipaddress.h		X
ns-2.33/tcp/send_queue.cc		X

Table 2. Modified source files.

As an example, packet.h defined various enumerated packet types in ns-2.27. The ns-BGP development added several additional packet types accordingly.

```
enum packet_t {
        <… other packet types …>
        PT_RTPROTO_BGP,
        PT_TCPMASTER,
        PT_PEERENTRY,
        <… other packet types …>
```

};

However, with the subsequent ns-2 development, ns-2 designers changed these enumerated packet types to unsigned integers with constant values in ns-2.33:

typedef unsigned int packet_t;
<… other packet types…>
static const packet_t PT_RTPROTO_BGP = 70;
static const packet_t PT_TCPMASTER = 71;
static const packet_t PT_PEERENTRY = 72;
<… other packet types …>

Consequently, further review and analysis of the source code was needed to identify appropriate constants for BGP specific packet types which did not overlap or create other “functional” issues. See Appendix 11.4 for further code integration changes.

The second stage involved the compilation modifications required to recompile the whole system accordingly. Specifically, there were several files as detailed in Table 2 that presented significant compilation errors. Again, a distinction is made between basic and moderate compilation errors using x and **X** respectively, where the latter indication also encompasses the time necessary to research the reported compiler error(s), the language syntax, and the context of the code logic where the error occurred. Nonetheless, these files exhibited significant dependencies on the compiler version furnished with the Linux distribution used at the time of ns-BGP development. The compiler versions for both environments are detailed in Table 3.

Compilers	ns-2.27	ns-2.33
gcc	3.3.3	4.1.2
g++	3.3.3	4.1.2

Table 3. Compiler versions.

One of the most complicated compilation issues was tied to the C++ Standard Template Library (STL). Specifically, within the ns-BGP TCP module, the logic used template library list containers to queue TCP data.

tcp/send_queue.cc: In member function ‘TcpData* SendQueue::get_data(int, int)’:
tcp/send_queue.cc:57: error: conversion from ‘int’ to non-scalar type ‘std::_List_iterator <SendData>’ requested
tcp/send_queue.cc:71: error: no match for ‘operator==‘ in ‘targetIterator == 0’
/usr/lib/gcc/i386-redhat-linux/4.1.2/../../../../include/c++/4.1.2/bits/stl_list.h:169: note: candidates are: bool std::_List_iterator<_Tp>::operator==(const std::_List_iterator <_Tp>&) const [with _Tp = SendData]
make: *** [tcp/send_queue.o] Error 1

While necessary code changes to overcome all the compilation issues was achieved with type casting logic bolded below,

57: list<SendData>::iterator targetIterator= **(list<SendData>::iterator)** NULL;
71: if(targetIterator == **(list<SendData>::iterator)** NULL)

this issue proved to be quite time consuming as it necessitated research into STL fundamentals to derive a suitable fix. Appendix 11.5 details the additional code compilation changes.

6 Validation

In order to validate the integration efforts, a significant amount of time was invested in provisioning a working native ns-BGP development environment which would subsequently serve as the baseline reference point in terms of BGP system functionality and outputs. Consequently, an ns-BGP environment running on the native ns-2 release that it was developed on (ns-2.26 / ns-2.27) was constructed. Within the ns-BGP 2.0 release, twelve Tcl test scripts were included to test the ns-2 BGP simulator functionality and BGP RFC compliancy.

Integration validation of this project is addressed in two ways. Firstly, the successful compilation of the detailed, manual code merges will validate the resulting programmatical syntax. Secondly, the functional validation of the resulting integration environment involves the execution and comparison of the ns-BGP specific Tcl scripts included in the ns-BPG 2.0 release.

Test Scripts	ns-2.27		ns-2.33		Equivalence	
	Output filename	Output file size	Output filename	Output filename size	File	Standard Out
drop-peer.tcl	drop-peer.nam	19055	drop-peer.nam	19055	✓	✓
drop-peer2.tcl	drop-peer2.nam	37213	drop-peer2.nam	37213	✓	✓
forwarding.tcl	forwarding.nam	3210797	forwarding.nam	3210797	✓	✓
	forwarding.out	1498410	forwarding.out	1498410	✓	✓
ibgp.tcl	ibgp.nam	24528	ibgp.nam	24528	✓	✓
keep-peer.tcl	keep-peer.nam	23892	keep-peer.nam	23892	✓	✓
propagation.tcl	propagation.nam	19779	propagation.nam	19779	✓	✓
reconnect.tcl	reconnect.nam	39793	reconnect.nam	39793	✓	✓
reflection.tcl	reflection.nam	87614	reflection.nam	87614	✓	✓
reflection2.tcl	reflection2.nam	85405	reflection2.nam	85405	✓	✓
route-distrib.tcl	route-distrib.nam	11668	route-distrib.nam	11668	✓	✓
select.tcl	select.nam	31338	select.nam	31338	✓	✓
withdrawals.tcl	withdrawals.nam	15060	withdrawals.nam	15060	✓	✓

Table 4. ns-BGP test script execution results.

By observing Table 4, the ns-BGP module in ns-2.33 generated equivalent output to ns-2.27 for all twelve test scripts. Specifically, each test script was executed in the native development and target integration environments and then compared using the UNIX diff utility. Moreover, the terminal output during the execution of each test script was also captured and compared across the environments; the output was identical for all twelve scripts.

While the validation process has confirmed syntactical and functional equivalence of the ns-BGP module between ns-2.27 and ns-2.33, it should be noted that the resulting ns-BGP functionality is that of the original ns-BGP implementation. Therefore, any original ns-BGP software bugs and/or computational inefficiencies will still exist. Moreover, any subsequent ns-BGP enhancements by academic and research communities are not included in this ns-2.33 ns-BGP release.

7 Related Work

Various research and commercial efforts have produced network BGP simulation support of varying degrees. Not surprisingly, the commercial simulation tool, OPNET Modeler [10] provides BGP support. The Modeler and ns-2 differ significantly in implementation and architecture thereby negating any potential porting efforts. SSFNet [9] provides BGP simulation support; in fact ns-BGP reflects the port of BGP from SSFNet. Since ns-2 continues to be heavily used in the academic and research communities, integrating ns-BGP in the current version of ns-2 yields the additional benefits of continued development and maintenance updates. SSFNet does not appear to have the same development interests. C-BGP [11] is a dedicated BGP "solver" application. It is used to compute the outcome of the BGP decision process, however it is a dedicated application rather than a widely configurable network simulation tool.

The Zebra BGP daemon [12] was ported to ns-2 around the same time ns-BGP was initially developed. It was written in C and provides features like BGP graceful restart, community support, and TCP MD5 authentication support. The last Zebra update was 2005. BGP++ [13] is a BGP module for ns-2 and GTNetS network simulators. It is actually a port of the Zebra bgp daemon and adapted to a C++ environment.

8 Future Work

The work conducted in this project focused on the integration of the ns-BGP 2.0 module developed in 2003 [1]. Subsequent enhancements to ns-BGP 2.0 could include multi-protocol extensions, policy routing, and route flap dampening [14].

Additionally, future efforts could integrate and test the updated ns-BGP release on Cygwin for Windows-based ns-2 simulations.

9 Conclusion

BGP is vital component of the Internet's routing infrastructure as it propagates NLRI between AS's. Recognizing the key role of BGP in today's Internet and the value of BGP network simulations in continued BGP research, this project has integrated the ns-BGP module originally developed for ns-2.26/ns-2.27 into

the current version of ns-2 (ns-2.33). By integrating the ns-BGP module into the current version of ns-2, BGP simulations can leverage off the continued ns-2 development and maintenance updates over the past five years.

This project has achieved its stated objective and excerised the available measures to demonstrate integration validity given the time constraints. While further efforts could be allocated to more rigorious integration testing between the two systems, the fact that the validation scripts generated identical output traces to the elaborate BGP specific test sequences, we gain a high degree of confidence in the resluting ns-BGP release integrity.

10 References

[1] T. D. Feng, R. Ballantyne, and Lj. Trajkovic, “Implementation of BGP in a network simulator,” *Applied Telecommunication Symposium, ATS '04*, Arlington, Virginia, Apr. 2004, pp. 149-154.

[2] I.Beijnum, *BGP*. Sebastopol, CA: O’Reilly & Associates, 2002.

[3] ns-2 [Online]. Available: http://www.isi.edu/nsnam/ns (May 2008).

[4] ns-2 manual [Online]. Available: http://www.isi.edu/nsnam/ns/doc/index.html (May 2008).

[5] Y. Rekhter and T. Li, *“A border gateway protocol 4 (BGP-4),”* RFC 1771, March 1995.

[6] R. Johnsonbaugh and J. Kalin, *Object-Oriented Programming in C++*. Englewood Cliffs, NJ: Prentice Hall, 1995.

[7] B. Welch, K. Jones, and J. Hobbs, *Practical Programming in Tcl and Tk 4/e*. Prentice Hall, 2003.

[8] BGP For Internet Service Providers [Online]. Available: http://www.cisco.com/public/cons/seminars/AfNOG3 (June 2008).

[9] SSFNet [Online]. Available: http://www.ssfnet.org/homePage.html (May 2008).

[10] OPNET BGP [Online]. Available: http://www.opnet.com (June 2008).

[11] C-BGP [Online]. Available: http://cbgp.info.ucl.ac.be/wiki/index.php (June 2008).

[12] GNU Zebra BGP daemon [Online]. Available: http://www.zebra.org/zebra/BGP.html#BGP (June 2008).

[13] BGP++ [Online]. Available: http://www.ece.gatech.edu/research/labs/MANIACS/BGP++ (June 2008).

[14] W. Shen and Lj. Trajkovic, “BGP route flap damping algorithms,” *Proc. SPECTS 2005*, Philadelphia, PA, July 2005, pp. 488-495.

[15] N. Laskovic and Lj. Trajkovic, ``BGP with an adaptive minimal route advertisement interval," *Proc. 25th IEEE Int. Performance, Computing, and Communications Conference*, Phoenix, AZ, April 2006, pp. 135-142.

11 APPENDICES

11.1 Acronyms

API	Application Programming Interface
AS	Autonomous System
BGP	Border Gateway Protocol
DARPA	Defense Advanced Research Projects Agency
DML	Domain Modeling Language
EBGP	Exterior Border Gateway Protocol
EGP	Exterior Gateway Protocol
FC	Fedora Core
FSM	Finite State Machine
GTNetS	Georgia Tech Network Simulator
HDD	Hard Disk Drive
IETF	Internet Engineering Task Force
IBGP	Interior Border Gateway Protocol
IGP	Interior Gateway Protocol
IGRP	Interior Gateway Routing Protocol
IP	Internet Protocol
ISP	Internet Service Providers
MD5	Message Digest 5
MED	Multiple Exit Discriminator
MPLS	Multi-Protocol Label Switching
ns-2	Network Simulator 2
ns-BGP	Network Simulator-Border Gateway Protocol
NLRI	Network Layer Reachability Information
OSPF	Open Shortest Path First
OTcl	Object Tool Command Language
QoS	Quality of Service
RFC	Request for Comments
RIP	Routing Information Protocol
RTP	Real Time Protocol
SP	Service Pack
SSFNet	Scalable Simulation Framework Network
SSFNet.OS.BGP4	SSFNet's BGP model
STL	Standard Template Library
Tcl	Tool Command Language
TCP	Transmission Control Protocol
Tk	Tool kit
UDP	User Datagram Protocol
VINT	Virtual Internetwork Testbed
VM	Virtual Machine
VPN	Virtual Private Network

11.2 Project Challenges

Various challenges have surfaced throughout this project. The primary challenge is the steep learning curve associated with learning the ns-2 internals along with the ns-BGP code module.

Moreover, the ability to install older Linux distributions to successfuly compile the ns-BGP module in its native ns-2 development release proved to be challenging due to limited hardware / software resources. To demonstrate, an attempt to compile ns-2.26 on Cygwin reported the following cryptic library errors:

```
gcc -pipe -c -O -Wall -Wconversion -Wno-implicit-int  -I./../generic -I. -
DHAVE_UNISTD_H=1 -DHAVE_LIMITS_H=1 -DHAVE_GETCWD=1 -DHAVE_OPENDIR=1 -
DHAVE_STRSTR=1 -DHAVE_STRTOL=1 -DHAVE_TMPNAM=1 -DHAVE_WAITPID=1 -DNO_VALUES_H=1
-DHAVE_UNISTD_H=1 -DHAVE_SYS_PARAM_H=1 -DUSE_TERMIOS=1 -DHAVE_SYS_TIME_H=1 -
DTIME_WITH_SYS_TIME=1 -DHAVE_TZNAME=1 -DHAVE_TIMEZONE_VAR=1 -DHAVE_ST_BLKSIZE=1
-DSTDC_HEADERS=1 -DNO_UNION_WAIT=1 -DNEED_MATHERR=1 -DHAVE_SIGNED_CHAR=1 -
DHAVE_SYS_IOCTL_H=1 -DSTATIC_BUILD=1          -DTCL_SHLIB_EXT=\"\"
./../unix/tclAppInit.c
gcc -pipe  tclAppInit.o -L ns-allinone-2.26/tcl8.3.2/unix -ltcl8.3  -lc \
                -o tclsh
fu000001.o:(.idata$2+0xc): undefined reference to `_libc_iname'
Info: resolving __timezone by linking to __imp___timezone (auto-import)
collect2: ld returned 1 exit status
make: *** [tclsh] Error 1
tcl8.3.2 make failed! Exiting ...
For problems with Tcl/Tk see http://www.scriptics.com
ns-allinone-2.26:$
```

An attempt to retry the same release as above but on FC 8 Linux revealed the following compilation errors:

```
g++ -c  -DNO_TK  -DNDEBUG -DUSE_SHM -DHAVE_LIBOTCL1_0A8 -DHAVE_OTCL_H -
DHAVE_LIBTK8_3 -DHAVE_TK_H -DHAVE_LIBTCL8_3 -DHAVE_TCL_H  -DSTDC_HEADERS=1 -
DHAVE_STRING_H=1 -DHAVE_SNPRINTF=1 -DSTDC_HEADERS=1 -DHAVE_STRTOQ=1 -
DHAVE_STRTOLL=1 -DHAVE_SYS_TYPES_H=1 -DHAVE_SYS_STAT_H=1 -DHAVE_STDLIB_H=1 -
DHAVE_STRING_H=1 -DHAVE_MEMORY_H=1 -DHAVE_STRINGS_H=1 -DHAVE_INTTYPES_H=1 -
DHAVE_STDINT_H=1 -DHAVE_UNISTD_H=1 -DSIZEOF_LONG=4 -DHAVE_INT64=1 -
DHAVE_TCL_H=1 -DHAVE_LIBTCL8_3=1 -DHAVE_TK_H=1 -DHAVE_LIBTK8_3=1 -
DHAVE_OTCL_H=1 -DHAVE_LIBOTCL1_0A8=1  -I. -I/home/whrudey/ns-allinone-
2.26/otcl-1.0a8 -I/home/whrudey/ns-allinone-2.26/include -I/home/whrudey/ns-
allinone-2.26/include -o Tcl.o Tcl.cc
tclcl-mappings.h: In static member function â:
tclcl-mappings.h:51: error: incomplete type â used in nested name specifier
tclcl-mappings.h:52: error: invalid use of undefined type â
tclcl-mappings.h:41: error: forward declaration of â
tclcl-mappings.h:57: error: invalid use of undefined type â
tclcl-mappings.h:41: error: forward declaration of â
make: *** [Tcl.o] Error 1
tclcl-1.0b13 make failed! Exiting ...
See http://www.isi.edu/nsnam/ns/ns-problems.html for problems
RFLab: /home/whrudey/ns-allinone-2.26$
```

Moreover, ns-2.27 compilation attempts on FC8 and FC4 reported these cryptic errors tied to incompatible Linux kernel header files:

```
g++ -c  -DTCP_DELAY_BIND_ALL -DNO_TK -DTCLCL_CLASSINSTVAR  -DNDEBUG -
DLINUX_TCP_HEADER -DUSE_SHM -DHAVE_LIBTCLCL -DHAVE_TCLCL_H -DHAVE_LIBOTCL1_8 -
DHAVE_OTCL_H -DHAVE_LIBTK8_4 -DHAVE_TK_H -DHAVE_LIBTCL8_4 -DHAVE_TCL_H  -
DHAVE_CONFIG_H -DNS_DIFFUSION -DSMAC_NO_SYNC -DCPP_NAMESPACE=std -
DUSE_SINGLE_ADDRESS_SPACE -Drng_test -I. -I/home/whrudey/ns-allinone-
2.27/tclcl-1.15 -I/home/whrudey/ns-allinone-2.27/otcl-1.8 -I/home/whrudey/ns-
allinone-2.27/include -I/home/whrudey/ns-allinone-2.27/include -
I/usr/include/pcap -I./tcp -I./sctp -I./common -I./link -I./queue -I./adc -
I./apps -I./mac -I./mobile -I./trace -I./routing -I./tools -I./classifier -
I./mcast -I./diffusion3/lib/main -I./diffusion3/lib -I./diffusion3/lib/nr -
I./diffusion3/ns -I./diffusion3/filter_core -I./asim/ -I./qs -o
diffusion3/ns/difftimer.o diffusion3/ns/difftimer.cc
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/stl_bvector.h: In member
function â:
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/stl_bvector.h:542: error:
expected unqualified-id before â token
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/stl_bvector.h: In member
function â:
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/stl_bvector.h:897: error:
expected unqualified-id before â token
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/vector.tcc: In member function
â:
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/vector.tcc:353: error: expected
unqualified-id before â token
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/vector.tcc: In member function
â:
/usr/lib/gcc/i386-redhat-
linux/4.1.2/../../../../include/c++/4.1.2/bits/vector.tcc:452: error: expected
unqualified-id before â token
make: *** [diffusion3/ns/difftimer.o] Error 1
```

With continued perseverance and significant effort allocated to executing unsuccessful “patch-like” fixes to FC8 and FC4 ns-2 environments, I was finally able to compile ns-2.26 and ns-2.27 on FC2. This breakthrough had a significant positive impact on my ability to validate the integration process.

11.3 ns-BGP 2.0 files

The initial ns-BGP 2.0 release archive contents are listed below:

```
ns-2.27/bgp/
ns-2.27/bgp/Util/
ns-2.27/bgp/Util/.deps/
ns-2.27/bgp/Util/.deps/stringmanip.Po
ns-2.27/bgp/Util/.deps/bitstring.Po
ns-2.27/bgp/Util/.deps/bit.Po
ns-2.27/bgp/Util/.deps/ipaddress.Po
ns-2.27/bgp/Util/bitstring.cc
ns-2.27/bgp/Util/radixtree.h
ns-2.27/bgp/Util/radixtreenode.h
ns-2.27/bgp/Util/stringmanip.h
```

```
ns-2.27/bgp/Util/stringmanip.cc
ns-2.27/bgp/Util/ipaddress.h
ns-2.27/bgp/Util/ipaddress.cc
ns-2.27/bgp/Util/bitstring.h
ns-2.27/bgp/Util/bit.h
ns-2.27/bgp/Path/.deps/atomicaggregate.Po
ns-2.27/bgp/Path/.deps/originatorid.Po
ns-2.27/bgp/Path/.deps/community.Po
ns-2.27/bgp/Path/.deps/med.Po
ns-2.27/bgp/Path/.deps/localpref.Po
ns-2.27/bgp/Path/.deps/segment.Po
ns-2.27/bgp/Path/.deps/aspath.Po
ns-2.27/bgp/Path/.deps/nexthop.Po
ns-2.27/bgp/Path/.deps/aggregator.Po
ns-2.27/bgp/Path/.deps/clusterlist.Po
ns-2.27/bgp/Path/.deps/origin.Po
ns-2.27/bgp/Path/.deps/attribute.Po
ns-2.27/bgp/Path/atomicaggregate.h
ns-2.27/bgp/Path/aggregator.cc
ns-2.27/bgp/Path/clusterlist.cc
ns-2.27/bgp/Path/aggregator.h
ns-2.27/bgp/Path/clusterlist.h
ns-2.27/bgp/Path/origin.h
ns-2.27/bgp/Path/community.cc
ns-2.27/bgp/Path/segment.cc
ns-2.27/bgp/Path/originatorid.h
ns-2.27/bgp/Path/med.cc
ns-2.27/bgp/Path/localpref.cc
ns-2.27/bgp/Path/attribute.cc
ns-2.27/bgp/Path/aspath.cc
ns-2.27/bgp/Path/originatorid.cc
ns-2.27/bgp/Path/nexthop.h
ns-2.27/bgp/Path/aspath.h
ns-2.27/bgp/Path/nexthop.cc
ns-2.27/bgp/Path/attribute.h
ns-2.27/bgp/Path/segment.h
ns-2.27/bgp/Path/med.h
ns-2.27/bgp/Path/community.h
ns-2.27/bgp/Path/origin.cc
ns-2.27/bgp/Path/atomicaggregate.cc
ns-2.27/bgp/Path/localpref.h
ns-2.27/bgp/Timing/.deps/
ns-2.27/bgp/Timing/.deps/timeoutmessage.Po
ns-2.27/bgp/Timing/.deps/scheduler.Po
ns-2.27/bgp/Timing/.deps/bgp_timer.Po
ns-2.27/bgp/Timing/.deps/timerhandler.Po
ns-2.27/bgp/Timing/mraiperpeertimer.cc
ns-2.27/bgp/Timing/bgp_timer.cc
ns-2.27/bgp/Timing/bgp_timer.h
ns-2.27/bgp/Timing/mraiperpeertimer.h
ns-2.27/bgp/Timing/mraitimer.h
ns-2.27/bgp/Timing/timeoutmessage.h
ns-2.27/bgp/Timing/timeoutmessage.cc
ns-2.27/bgp/Timing/mraitimer.cc
ns-2.27/bgp/Comm/.deps/
ns-2.27/bgp/Comm/.deps/notificationmessage.Po
ns-2.27/bgp/Comm/.deps/startstopmessage.Po
ns-2.27/bgp/Comm/.deps/openmessage.Po
ns-2.27/bgp/Comm/.deps/keepalivemessage.Po
ns-2.27/bgp/Comm/.deps/updatemessage.Po
ns-2.27/bgp/Comm/.deps/message.Po
ns-2.27/bgp/Comm/.deps/transportmessage.Po
ns-2.27/bgp/Comm/updatemessage.cc
```

```
ns-2.27/bgp/Comm/bgpmessage.cc
ns-2.27/bgp/Comm/openmessage.cc
ns-2.27/bgp/Comm/notificationmessage.h
ns-2.27/bgp/Comm/transportmessage.h
ns-2.27/bgp/Comm/keepalivemessage.cc
ns-2.27/bgp/Comm/notificationmessage.cc
ns-2.27/bgp/Comm/startstopmessage.h
ns-2.27/bgp/Comm/keepalivemessage.h
ns-2.27/bgp/Comm/startstopmessage.cc
ns-2.27/bgp/Comm/transportmessage.cc
ns-2.27/bgp/Comm/updatemessage.h
ns-2.27/bgp/Comm/openmessage.h
ns-2.27/bgp/Comm/bgpmessage.h
ns-2.27/bgp/route.cc
ns-2.27/bgp/route.h
ns-2.27/bgp/classifier-ipv4.cc
ns-2.27/bgp/peer-entry.h
ns-2.27/bgp/classifier-ipv4src.h
ns-2.27/bgp/adjribout.cc
ns-2.27/bgp/adjribin.cc
ns-2.27/bgp/global.h
ns-2.27/bgp/classifier-ipv4.h
ns-2.27/bgp/ribelement.h
ns-2.27/bgp/routeinfo.cc
ns-2.27/bgp/rtProtoBGP.h
ns-2.27/bgp/rtProtoBGP.cc
ns-2.27/bgp/locrib.h
ns-2.27/bgp/locrib.cc
ns-2.27/bgp/routeinfo.h
ns-2.27/bgp/classifier-ipv4src.cc
ns-2.27/bgp/ribelement.cc
ns-2.27/bgp/adjribin.h
ns-2.27/bgp/adjribout.h
ns-2.27/bgp/peer-entry.cc
ns-2.27/bgp/Readme.txt
ns-2.27/tcl/bgp/test/
ns-2.27/tcl/bgp/test/route-distrib.tcl
ns-2.27/tcl/bgp/test/reflection2.tcl
ns-2.27/tcl/bgp/test/drop-peer2.tcl
ns-2.27/tcl/bgp/test/propagation.tcl
ns-2.27/tcl/bgp/test/keep-peer.tcl
ns-2.27/tcl/bgp/test/forwarding.tcl
ns-2.27/tcl/bgp/test/withdrawals.tcl
ns-2.27/tcl/bgp/test/drop-peer.tcl
ns-2.27/tcl/bgp/test/reflection.tcl
ns-2.27/tcl/bgp/test/ibgp.tcl
ns-2.27/tcl/bgp/test/reconnect.tcl
ns-2.27/tcl/bgp/test/select.tcl
ns-2.27/tcl/bgp/ns-bgp-peerentry.tcl
ns-2.27/tcl/bgp/ns-tcpmaster.tcl
ns-2.27/tcl/bgp/ns-rtProtoBGP.tcl
ns-2.27/tcl/bgp/ns-bgp-node.tcl
ns-2.27/tcp/receive_queue.cc
ns-2.27/tcp/send_queue.h
ns-2.27/tcp/send_queue.cc
ns-2.27/tcp/continuation.h
ns-2.27/tcp/tcp_socket.cc
ns-2.27/tcp/tcp_master.h
ns-2.27/tcp/tcp_data.h
ns-2.27/tcp/receive_queue.h
ns-2.27/tcp/tcp_socket.h
ns-2.27/tcp/tcp_master.cc
ns-2.27/tcp/tcp_data.cc
```

11.4 Code Integration Changes

The following section details the code integration changes necessary to migrate ns-BGP to ns-2.33 in this project.

diff -w 233_original/Makefile.in 233_merged_final/Makefile.in

```
318a319,332
>      tcp/tcp_data.o tcp/receive_queue.o tcp/send_queue.o tcp/tcp_master.o
tcp/tcp_socket.o \
>      bgp/Util/bitstring.o bgp/Util/ipaddress.o bgp/Util/stringmanip.o \
>      bgp/Comm/bgpmessage.o bgp/Comm/keepalivemessage.o
bgp/Comm/notificationmessage.o \
>      bgp/Comm/openmessage.o bgp/Comm/startstopmessage.o
bgp/Comm/transportmessage.o \
>      bgp/Comm/updatemessage.o \
>      bgp/Timing/bgp_timer.o bgp/Timing/timeoutmessage.o bgp/Timing/mraitimer.o
bgp/Timing/mraiperpeertimer.o \
>      bgp/Path/aggregator.o bgp/Path/aspath.o bgp/Path/atomicaggregate.o \
>      bgp/Path/attribute.o bgp/Path/clusterlist.o bgp/Path/community.o \
>      bgp/Path/localpref.o bgp/Path/med.o bgp/Path/nexthop.o \
>      bgp/Path/originatorid.o bgp/Path/origin.o bgp/Path/segment.o \
>      bgp/route.o bgp/routeinfo.o \
>      bgp/ribelement.o bgp/adjribin.o bgp/adjribout.o bgp/locrib.o \
>         bgp/classifier-ipv4.o bgp/classifier-ipv4src.o bgp/peer-entry.o
bgp/rtProtoBGP.o \
509a524,528
>      tcl/bgp/ns-bgp-node.tcl \
>      tcl/bgp/ns-bgp-peerentry.tcl \
>      tcl/bgp/ns-rtProtoBGP.tcl \
>      tcl/bgp/ns-tcpmaster.tcl \
```

diff -w 233_original/node.cc 233_merged_final/node.cc

```
141a142,148
> // Merged by Will Hrudey
>         // Modified by Tony Feng for BGP.
>         if (strcmp(argv[1], "as") == 0) {
>                   as_num_ = atoi(argv[2]);
>                         return TCL_OK;
>         }
>         // End Tony Feng
216a224,230
> // Merged by Will Hrudey
>             } // Modified by Tony Feng for BGP.
>                   else if (strcmp(argv[1], "add-AS-neighbor") == 0) {
>                   Node * node = (Node *)TclObject::lookup(argv[2]);
>                   if (node == 0) {
>                         tcl.resultf("Invalid node %s", argv[2]);
>                                 return (TCL_ERROR);
217a232,234
>                   ASnbs.push_back(node);
>                   return TCL_OK;
>             } // End Tony Feng.
```

diff -w 233_original/node.h 233_merged_final/node.h

```
61a62,63
> // Merged by Will Hrudey
```

```
> #include <list>
132a135,136
> // Merged by Will Hrudey
>      inline int as_number() { return as_num_; } //Added by Tony Feng
157a162,164
> // Merged by Will Hrudey
>         //AS neighbor list for BGP autoconfig, added by Tony Feng
>         list<Node*> ASnbs;
170a178,179
> // Merged by Will Hrudey
>      int as_num_;     // Added by Tony Feng for BGP
```

diff -w 233_original/ns-default.tcl 233_merged_final/ns-default.tcl

```
1346a1347,1365
> # Merged by Will Hrudey
> # Added by Tony Feng for BGP
> Agent/rtProto/BGP set connretry_interval_ 120
> Agent/rtProto/BGP set masoi_ 15
> Agent/rtProto/BGP set cluster_num 0
> Agent/rtProto/BGP set bgp_id_ 0
> Agent/rtProto/BGP set as_num_ 0
> Agent/rtProto/BGP set auto_config_ false
> Agent/rtProto/BGP set preference_  80
>
> # PeerEntry
> Agent/PeerEntry set ipaddr_ 0
> Agent/PeerEntry set as_num_ 0
> Agent/PeerEntry set bgp_id_ 0
> Agent/PeerEntry set return_ipaddr_ 0
> Agent/PeerEntry set hold_time_ 90
> Agent/PeerEntry set keep_alive_interval_ 30
> Agent/PeerEntry set mrai_ 30
> # End Tony Feng
```

diff -w 233_original/ns-lib.tcl 233_merged_final/ns-lib.tcl

```
231a232,240
> # Mergedy by Will Hrudey
> # Added by Tony Feng for BGP
> source ../bgp/ns-bgp-node.tcl
> source ../bgp/ns-rtProtoBGP.tcl
> source ../bgp/ns-bgp-peerentry.tcl
>
> #TCPMaster
> source ../bgp/ns-tcpmaster.tcl
> # End Tony Feng
397a407,418
> # Merged by Will Hrudey
> # Added by Tony Feng for BGP
> Simulator instproc BGP { val } {
>      if { $val == "ON" } {
>             Node enable-module BGP
>             Node disable-module Base
>      } else {
>             Node disable-module BGP
>             Node enable-module Base
>      }
> }
> # End Tony Feng
1104a1126,1133
> # Merged by Will Hrudey
>      # Added by Tony Feng for BGP.
```

```
>      if { [$n1 set as_num_] != [$n2 set as_num_] } {
>      # n1 and n2 reside in different AS, add a route to n2 in n1's classifier.
>             $n1 add-route [$n2 set address_] [[set link_($sid:$did)] set
head_]
>             $n1 cmd add-AS-neighbor $n2
>      }
>      # End Tony Feng.
```

diff -w 233_original/ns-node.tcl 233_merged_final/ns-node.tcl

```
69c69
<                    nodetype_ multiPath_ ns_ rtnotif_ ptnotif_
---
>                    nodetype_ multiPath_ ns_ rtnotif_ ptnotif_ as_num_
74a75,76
> # Merged by Will Hrudey
>       # Modified by Tony Feng for BGP
76c78,88
<             set address_ [lindex $args 0]
---
>                    if {[llength $args] == 1} {
>             set arg_0 [lindex $args 0]
>                    if { [scan $arg_0 "%s" ] != -1 } {
>                           # create node with arg_0 of [as_num:ipaddr] format
>                           $self parse-addr $arg_0
>                    } else {
>                           # create node with arg_0 of int value
>                           set address_ $arg_0
>                           set as_num_ 0
>                    }
>                    }
78a91
>             set as_num_ 0
79a93
>      # End Tony Feng
```

diff -w 233_original/packet.h 233_merged_final/packet.h

```
102a103,106
> // Merged by Will Hrudey
> static const packet_t PT_RTPROTO_BGP = 70;  // For bgp implementation, added
by Tony Feng
> static const packet_t PT_TCPMASTER = 71;
> static const packet_t PT_PEERENTRY = 72;     // end Tony Feng
302a307,310
> // Merged by Will Hrudey
>             name_[PT_RTPROTO_BGP]= "rtProtoBGP"; // For bgp implementation,
added by Tony Feng
>             name_[PT_PEERENTRY]= "PeerEntry";
>             name_[PT_TCPMASTER]= "tcpmaster"; // end Tony Feng
```

diff -w 233_original/route.cc 233_merged_final/route.cc

```
395a396,405
> // Merged by Will Hrudey
>         // Modified by Tony Feng. check if src and dst come from the same as.
>         // Note that index = nodeid +1.
>         Tcl& tcl = Tcl::instance();
>      tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",src-1);
>      int as_num_src = atoi(tcl.result());
>         tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",dst-
1);
>      int as_num_dst = atoi(tcl.result());
```

```
>         if (as_num_src == as_num_dst)
>         // End Tony Feng.
401a412,420
> // Merged by Will Hrudey
>         // Modified by Tony Feng. check if src and dst come from the same AS.
>         Tcl& tcl = Tcl::instance();
>      tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",src-1);
>      int as_num_src = atoi(tcl.result());
>         tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",dst-
1);
>      int as_num_dst = atoi(tcl.result());
>         if (as_num_src == as_num_dst) {
>         // End Tony Feng.
404a424
> }
```

diff -w 233_original/rq.cc 233_merged_final/rq.cc

```
300a301
> // Merged by Will Hrudey
302c303
< ReassemblyQueue::add(TcpSeq start, TcpSeq end, TcpFlag tiflags, RqFlag
rqflags)
---
> ReassemblyQueue::add(TcpSeq start, TcpSeq end, TcpFlag tiflags, RqFlag
rqflags, AppData* data)
330a332,333
> // Merged by Will Hrudey
>                 head_->data = data;                //Added by Zheng Wang for
BGP
502a506,508
> // Merged by Will Hrudey
>                 n->data = data; //Added by Zheng Wang for BGP
>
531a538,539
> // Merged by Will Hrudey
>              toReceiveQueue->enqueue((TcpData*)(q->data)); //Added by Zheng
Wang for BGP
```

diff -w 233_original/rq.h 233_merged_final/rq.h

```
70a71,72
> // Merged by Will Hrudey
> #include "receive_queue.h" //Added by Zheng Wang for BGP
101a104,105
> // Merged by Will Hrudey
>                 AppData* data;          // Added by Zheng Wang for BGP
108c112,113
<      int add(TcpSeq sseq, TcpSeq eseq, TcpFlag pflags, RqFlag rqflags = 0);
---
> // Merged by Will Hrudey
>      int add(TcpSeq sseq, TcpSeq eseq, TcpFlag pflags, RqFlag rqflags = 0,
AppData* data = 0); //Modified by Zheng Wang for BGP
123a129,130
> // Merged by Will Hrudey
>      void connRevQueue(ReceiveQueue* revQueue){toReceiveQueue = revQueue;}
//Added by Zheng Wang for BGP
145a153,155
> // Merged by Will Hrudey
>         ReceiveQueue* toReceiveQueue;  //Added by Zheng Wang for BGP
>
```

diff -w 233_original/rtmodule.cc 233_merged_final/rtmodule.cc

```
144a145,155
> // Merged by Will Hrudey
> // Added by Tony Feng for BGP.
> static class BGPRoutingModuleClass : public TclClass {
> public:
>      BGPRoutingModuleClass() : TclClass("RtModule/BGP") {}
>      TclObject* create(int, const char*const*) {
>            return ( new BGPRoutingModule );
>      }
> } class_bgp_module;
> // End Tony Feng.
>
511a523,546
> // Merged by Will Hrudey
> // Added by Tony Feng for BGP.
> BGPRoutingModule::BGPRoutingModule() { }
>
> int BGPRoutingModule::command(int argc, const char*const* argv) {
>      Tcl& tcl = Tcl::instance();
>      if ( argc == 3 ) {
>            if ( strcmp(argv[1], "route-notify") == 0 ) {
>              Node *node = (Node *)(TclObject::lookup(argv[2]));
>                   if (node == NULL) {
>                         tcl.add_errorf("Invalid node object %s", argv[2]);
>                         return TCL_ERROR;
>                   }
>                   if (node != n_) {
>                         tcl.add_errorf("Node object %s different from n_",
argv[2]);
>                         return TCL_ERROR;
>                   }
>                   n_->route_notify(this);
>                   return TCL_OK;
>            }
>      }
>      return (RoutingModule::command(argc, argv));
> }
> // End Tony Feng.
```

diff -w 233_original/rtmodule.h 233_merged_final/rtmodule.h

```
69a70,74
> // Merged by Will Hrudey
> // Added by Tony Feng for BGP.
> class IPv4Classifier;
> class rtProtoBGP;
> // End Tony Feng.
180a186,197
> // Merged by Will Hrudey
> // Added by Tony Feng for BGP.
> class BGPRoutingModule : public RoutingModule {
> public:
>         BGPRoutingModule();
>         virtual const char* module_name() const {return "BGP";}
>         virtual int command (int argc, const char* const * argv);
> protected:
>         IPv4Classifier *classifier_;
>         rtProtoBGP *bgp_agent_;
> };
> // End Tony Feng.
```

diff -w 233_original/scoreboard-rq.cc 233_merged_final/scoreboard-rq.cc

```
87c87,88
<             rq_.add(tcph->sa_left(i), tcph->sa_right(i), 0);
---
> // Merged by Will Hrudey
>             rq_.add(tcph->sa_left(i), tcph->sa_right(i), 0, NULL);
```

diff -w 233_original/simulator.cc 233_merged_final/simulator.cc

```
192c192,197
<                                 sprintf(tmp, "%d", j);
---
> // Merged by Will Hrudey
>                                 // Modified by Tony Feng. We use the node
address instead of node_id.
>                                         Tcl::instance().evalf("[Simulator
instance] get-node-by-id %d",j);
>                                         Node * node = (Node*)
TclObject::lookup(Tcl::instance().result());
>                                         sprintf(tmp, "%d", node->address());
>                                         // End Tony Feng.
```

diff -w 233_original/tcp-full.cc 233_merged_final/tcp-full.cc

```
343a344,365
> // Merged by Will Hrudey
>   /*
>    * send a string of nBytes, added by Zheng Wang for BGP
>    */
> void
> FullTcpAgent::advance_bytes(int nBytes, const char* const data, Continuation*
caller)
> {
>       if (writeCont != NULL){
>         printf("write error - socket already in blocking write\n");
>         if (caller != NULL)
>           caller->failure();
>         return;
>       }
>       if(toSendQueue==NULL)  {
>         toSendQueue = new SendQueue();
>       }
>       if (data!=NULL) {
>         writeCont = caller;
>         toSendQueue->enqueue(nBytes,data);
>       }
>       advance_bytes(nBytes);
> }
466a489,491
> // Merged by Will Hrudey
>                 if(mySocket)        // added by Tony Feng, informs the socket
that tcp closed successfully
>                     mySocket->disconnected();
470a496,498
> // Merged by Will Hrudey
>                 if(mySocket)        // added by Tony Feng, informs the socket
that tcp closed successfully
>                     mySocket->disconnected();
717a746,748
> // Merged by Will Hrudey
>       AppData* data = pkt->userdata();   //Added by Zheng Wang for BGP
>
726,727c757,758
```

```
<
<      flags = rq_.add(start, end, tiflags, 0);
---
> // Merged by Will Hrudey
>      flags = rq_.add(start, end, tiflags, 0, data); //Modified by Zheng Wang
for BGP
750a782,825
> // Merged by Will Hrudey
> //Added by Zheng Wang for BGP
> void FullTcpAgent::read(char* buffer, int nbytes, Continuation* caller)
> {
>      if (readCont != NULL){
>        printf("read error - socket already in blocking read\n");
>        if (caller != NULL)
>          caller->failure();
>        return;
>      }
>
>      // if requested data is in the buffer, get it from buffer
>      // We use the fact that a data object arrives in the first TCP segment.
>      if(nbytes <= dataReceived) {
>        dataReceived -= nbytes;
>        if(toReceiveQueue->is_empty()) {
>            printf("receive queue is empty, exit\n");
>          return;
>        }
>        toReceiveQueue->retrieve_data(nbytes,buffer);
>          caller->success();
>      } else {
>        inbuffer = buffer;
>        readCont = caller;
>        readSize = nbytes;
>      }
>  }
>
> void FullTcpAgent::recvBytes(int bytes)
>  {
>      dataReceived+= bytes;
>      if((readCont != NULL) && (dataReceived >= readSize)) {
>          toReceiveQueue->retrieve_data(readSize,inbuffer);
>          mySocket->appCallWaiting = false;
>        //mySocket->app_call_waiting = NULL;
>        dataReceived -= readSize;
>        Continuation* rc = readCont;
>        readCont = NULL;
>        rc->success();
>
>      }
>      Agent::recvBytes(bytes);
>  }
> // End Zheng Wang
909a985,996
> // Merged by Will Hrudey
>      //Added by Zheng Wang for BGP
>            //Set data field
>            if(toSendQueue)
>            {
>                  if(!toSendQueue->is_empty())
>                  {
>                        TcpData* pData = toSendQueue-
>get_data(seqno,datalen);
>                        p->setdata(pData);
>                  }
```

```
>            }
>      //End Zheng Wang
1285a1373,1381
> // Merged by Will Hrudey
>              // Added by Tony Feng for Socket::wirte()
>                 if(writeCont != NULL) {
>                    Continuation *  app_call_waiting = writeCont;
>                    writeCont = NULL;
>                    mySocket->appCallWaiting = false;
>                    app_call_waiting->success();
>                 }
>                 // End Tony Feng
1625a1722,1723
> // Merged by Will Hrudey
>                   toReceiveQueue->enqueue((TcpData*)pkt->userdata()); //
Added by Zheng Wang for BGP
1707a1806,1813
> // Merged by Will Hrudey
>             //Modified by Zheng Wang
>             if( mySocket!=NULL && mySocket->isListening) {
>                         hdr_ip* iph1 = hdr_ip::access(pkt);
>                         //Received SYN for listening tcp, we create a new
reading socket.
>                     tcpMaster->newInComing(pkt,mySocket);
>                         return;
>             }  else {
1708a1815,1816
>                 }
>             // End Zheng Wang
1795a1904,1908
> // Merged by Will Hrudey
>                                   //Added by Zheng Wang
>                                   if (mySocket)
>                                 mySocket->connected();
>                                   //End Zheng Wang
2027a2141,2142
> // Merged by Will Hrudey
>                           mySocket->listeningSocket->addConnection(mySocket);
//Added by Zheng Wang for BGP
2278a2394,2399
> // Merged by Will Hrudey
>                                   // Added by Tony Feng for BGP
>                                   if(mySocket)  {
>                                        mySocket->disconnected(); // Informs the
socket that tcp closed successfully
>                                   }
>                                   // End Tony Feng
2291a2413,2418
> // Merged by Will Hrudey
>                                   // Added by Tony Feng for BGP
>                                   if(mySocket)  {
>                                        mySocket->disconnected(); // Informs the
socket that tcp closed successfully
>                                   }
>                                   // End Tony Feng
2579c2706,2707
<      if (state_ == TCPS_LISTEN) {
---
> // Merged by Will Hrudey
>      if (state_ == TCPS_LISTEN && (mySocket==NULL || !mySocket->isListening))
{ // Modified by Tony Feng for BGP
```

diff -w 233_original/tcp-full.h 233_merged_final/tcp-full.h

```
41a42,49
> // Merged by Will Hrudey
> //Added by Zheng Wang
> #include "send_queue.h"
> #include "receive_queue.h"
> #include "tcp_socket.h"
> #include "tcp_master.h"
> #include "continuation.h"
> //End Zheng Wang
116a125,126
> // Merged by Will Hrudey
> friend class TcpMaster;
123c133,142
<             last_state_(TCPS_CLOSED), rq_(rcv_nxt_), last_ack_sent_(-1) { }
---
> // Merged by Will Hrudey
>             last_state_(TCPS_CLOSED), rq_(rcv_nxt_), last_ack_sent_(-1)
>                  {   // Modified by Zheng Wang for BGP
>                        toSendQueue = new SendQueue();
>                   toReceiveQueue = new ReceiveQueue();
>                        rq_.connRevQueue(toReceiveQueue);
>                   dataReceived = 0;
>                      readCont = NULL;
>                      writeCont = NULL;
>                    }
125c144,151
<      ~FullTcpAgent() { cancel_timers(); rq_.clear(); }
---
>      ~FullTcpAgent()
>                 {    cancel_timers();
>                      rq_.clear();
>                      if(toSendQueue)
>                     delete toSendQueue;
>                  if(toReceiveQueue)
>                      delete toReceiveQueue;
>                 }    // End Zheng Wang
130a157,162
> // Merged by Will Hrudey
>      // Added by Zheng Wang for BGP
>      void advance_bytes(int nBytes, const char* const data, Continuation*
caller);
>      void recvBytes(int bytes);  // Overrides Agent's recvBytes();
>      void read(char* buffer, int nbytes, Continuation* caller);
>      // End Zheng Wang
134a167,172
> // Merged by Will Hrudey
>      // Added by Zheng Wang
>      ReceiveQueue* getRevQueue() {return toReceiveQueue;}
>         TcpSocket* mySocket;
>         TcpMaster* tcpMaster;
>      // End Zheng Wang
240a279,289
> // Merged by Will Hrudey
>      // Added by Zheng Wang for BGP
>      SendQueue* toSendQueue;
>      ReceiveQueue* toReceiveQueue;
>      int dataReceived;
>      Continuation* readCont;
>         Continuation* writeCont;
>      int readSize;
>         int writeSize;
>      char* inbuffer;
```

```
>      // End Zheng Wang
```

11.5 Code Compilation Changes

The following section details the code changes necessary to successfully compile ns-BGP in ns-2.33 after the code integration phase.

diff -w 233_compilation_issues/ipaddress.cc 233_compilation_final/ipaddress.cc

```
18a19,20
> // Modified by Will Hrudey
> #include <math.h>
```

diff -w 233_compilation_issues/ipaddress.h 233_compilation_final/ipaddress.h

```
49c49,51
<   boolVector IPaddress::str2bin(string str);
---
>   // Modified by Will Hrudey
>   //boolVector IPaddress::str2bin(string str);
>   boolVector str2bin(string str);
```

diff -w 233_compilation_issues/send_queue.cc
233_compilation_final/send_queue.cc

```
55c55,59
<      list<SendData>::iterator targetIterator=NULL;
---
>
>      // Modified by Will Hrudey
>      // list<SendData>::iterator targetIterator= NULL;
>      list<SendData>::iterator targetIterator= (list<SendData>::iterator) NULL;
>
66c70,72
<      if(targetIterator == NULL)
---
>      // Modified by Will Hrudey
>      // if(targetIterator == NULL)
>      if(targetIterator == (list<SendData>::iterator) NULL)
```

11.6 ns-BGP Patch File

The following section details the resulting project ns-BGP patch file required to patch the ns-BGP 2.0 release in the ns-2.33 environment:

```
diff -rc ns-2.33.orig/common/node.cc ns-2.33/common/node.cc
*** ns-2.33.orig/common/node.cc  2008-03-31 19:00:25.000000000 -0700
--- ns-2.33/common/node.cc 2008-07-26 20:38:10.000000000 -0700
***************
*** 139,144 ****
--- 139,151 ----
  Node::command(int argc, const char*const* argv)
  {
       Tcl& tcl = Tcl::instance();
+ // Merged by Will Hrudey
+         // Modified by Tony Feng for BGP.
+         if (strcmp(argv[1], "as") == 0) {
+                  as_num_ = atoi(argv[2]);
```

```
+                         return TCL_OK;
+         }
+         // End Tony Feng
       if (argc == 2) {
  #ifdef HAVE_STL
             // Mods for Nix-Vector Routing
***************
*** 214,220 ****
                     }
                     addNeighbor(node);
                     return TCL_OK;
!            }
       }
       return ParentNode::command(argc,argv);
  }
--- 221,237 ----
                     }
                     addNeighbor(node);
                     return TCL_OK;
! // Merged by Will Hrudey
!            } // Modified by Tony Feng for BGP.
!                    else if (strcmp(argv[1], "add-AS-neighbor") == 0) {
!                    Node * node = (Node *)TclObject::lookup(argv[2]);
!                    if (node == 0) {
!                           tcl.resultf("Invalid node %s", argv[2]);
!                                  return (TCL_ERROR);
!                    }
!                    ASnbs.push_back(node);
!                    return TCL_OK;
!            } // End Tony Feng.
       }
       return ParentNode::command(argc,argv);
  }
diff -rc ns-2.33.orig/common/node.h ns-2.33/common/node.h
*** ns-2.33.orig/common/node.h   2008-03-31 19:00:25.000000000 -0700
--- ns-2.33/common/node.h  2008-07-26 20:38:14.000000000 -0700
***************
*** 59,64 ****
--- 59,66 ----
  #include "energy-model.h"
  #include "location.h"
  #include "rtmodule.h"
+ // Merged by Will Hrudey
+ #include <list>

  class NixNode;
  class LinkHead;
***************
*** 130,135 ****
--- 132,139 ----

       inline int address() { return address_;}
       inline int nodeid() { return nodeid_;}
+ // Merged by Will Hrudey
+      inline int as_number() { return as_num_; } //Added by Tony Feng
       inline bool exist_namchan() const { return (namChan_ != 0); }

       virtual int command(int argc, const char*const* argv);
***************
*** 155,160 ****
--- 159,167 ----
       void addNeighbor(Node *node);
```

```
       static Node* get_node_by_address(nsaddr_t);
+ // Merged by Will Hrudey
+         //AS neighbor list for BGP autoconfig, added by Tony Feng
+         list<Node*> ASnbs;

       //routines for supporting routing
       void route_notify (RoutingModule *rtm);
***************
*** 168,173 ****
--- 175,182 ----
       LIST_ENTRY(Node) entry;  // declare list entry structure
       int address_;
       int nodeid_;         // for nam use
+ // Merged by Will Hrudey
+      int as_num_;     // Added by Tony Feng for BGP

       // Nam tracing facility
          Tcl_Channel namChan_;
diff -rc ns-2.33.orig/common/packet.h ns-2.33/common/packet.h
*** ns-2.33.orig/common/packet.h 2008-03-31 19:00:25.000000000 -0700
--- ns-2.33/common/packet.h      2008-07-26 20:38:24.000000000 -0700
***************
*** 100,105 ****
--- 100,109 ----
  static const packet_t PT_RTCP = 14;
  static const packet_t PT_RTP = 15;
  static const packet_t PT_RTPROTO_DV = 16;
+ // Merged by Will Hrudey
+ static const packet_t PT_RTPROTO_BGP = 70;  // For bgp implementation, added
by Tony Feng
+ static const packet_t PT_TCPMASTER = 71;
+ static const packet_t PT_PEERENTRY = 72;     // end Tony Feng
  static const packet_t PT_CtrMcast_Encap = 17;
  static const packet_t PT_CtrMcast_Decap = 18;
  static const packet_t PT_SRM = 19;
***************
*** 300,305 ****
--- 304,313 ----
             name_[PT_RTCP]= "rtcp";
             name_[PT_RTP]= "rtp";
             name_[PT_RTPROTO_DV]= "rtProtoDV";
+ // Merged by Will Hrudey
+            name_[PT_RTPROTO_BGP]= "rtProtoBGP"; // For bgp implementation,
added by Tony Feng
+            name_[PT_PEERENTRY]= "PeerEntry";
+            name_[PT_TCPMASTER]= "tcpmaster"; // end Tony Feng
             name_[PT_CtrMcast_Encap]= "CtrMcast_Encap";
             name_[PT_CtrMcast_Decap]= "CtrMcast_Decap";
             name_[PT_SRM]= "SRM";
diff -rc ns-2.33.orig/common/simulator.cc ns-2.33/common/simulator.cc
*** ns-2.33.orig/common/simulator.cc    2008-03-31 19:00:25.000000000 -0700
--- ns-2.33/common/simulator.cc  2008-07-26 20:38:33.000000000 -0700
***************
*** 189,195 ****
                          nh = rtobject_->lookup_flat(i, j);
                          if (nh >= 0) {
                                 NsObject *l_head = get_link_head(nodelist_[i],
nh);
!                                sprintf(tmp, "%d", j);
                                 nodelist_[i]->add_route(tmp, l_head);
                          }
                  }
--- 189,200 ----
```

```
                            nh = rtobject_->lookup_flat(i, j);
                            if (nh >= 0) {
                                   NsObject *l_head = get_link_head(nodelist_[i],
nh);
! // Merged by Will Hrudey
!                                  // Modified by Tony Feng. We use the node
address instead of node_id.
!                                          Tcl::instance().evalf("[Simulator
instance] get-node-by-id %d",j);
!                                          Node * node = (Node*)
TclObject::lookup(Tcl::instance().result());
!                                          sprintf(tmp, "%d", node->address());
!                                          // End Tony Feng.
                                   nodelist_[i]->add_route(tmp, l_head);
                            }
                     }
diff -rc ns-2.33.orig/Makefile.in ns-2.33/Makefile.in
*** ns-2.33.orig/Makefile.in     2008-03-31 19:00:08.000000000 -0700
--- ns-2.33/Makefile.in    2008-07-26 20:31:54.000000000 -0700
***************
*** 18,23 ****
--- 18,25 ----
  #  MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
  #
  # @(#) $Header: 2002/10/09 15:34:11
+ # Merged code @ lines 320 & 524 by Will Hrudey
+ #

  #
  # Various configurable paths (remember to edit Makefile.in, not Makefile)
***************
*** 316,321 ****
--- 318,336 ----
       mcast/classifier-lms.o mcast/lms-agent.o mcast/lms-receiver.o \
       mcast/lms-sender.o \
       queue/delayer.o \
+      tcp/tcp_data.o tcp/receive_queue.o tcp/send_queue.o tcp/tcp_master.o
tcp/tcp_socket.o \
+      bgp/Util/bitstring.o bgp/Util/ipaddress.o bgp/Util/stringmanip.o \
+      bgp/Comm/bgpmessage.o bgp/Comm/keepalivemessage.o
bgp/Comm/notificationmessage.o \
+      bgp/Comm/openmessage.o bgp/Comm/startstopmessage.o
bgp/Comm/transportmessage.o \
+      bgp/Comm/updatemessage.o \
+      bgp/Timing/bgp_timer.o bgp/Timing/timeoutmessage.o bgp/Timing/mraitimer.o
bgp/Timing/mraiperpeertimer.o \
+      bgp/Path/aggregator.o bgp/Path/aspath.o bgp/Path/atomicaggregate.o \
+      bgp/Path/attribute.o bgp/Path/clusterlist.o bgp/Path/community.o \
+      bgp/Path/localpref.o bgp/Path/med.o bgp/Path/nexthop.o \
+      bgp/Path/originatorid.o bgp/Path/origin.o bgp/Path/segment.o \
+      bgp/route.o bgp/routeinfo.o \
+      bgp/ribelement.o bgp/adjribin.o bgp/adjribout.o bgp/locrib.o \
+         bgp/classifier-ipv4.o bgp/classifier-ipv4src.o bgp/peer-entry.o
bgp/rtProtoBGP.o \
       xcp/xcpq.o xcp/xcp.o xcp/xcp-end-sys.o \
       wpan/p802_15_4csmaca.o wpan/p802_15_4fail.o \
       wpan/p802_15_4hlist.o wpan/p802_15_4mac.o \
***************
*** 507,512 ****
--- 522,531 ----
       tcl/lib/ns-srcrt.tcl \
       tcl/mcast/ns-lms.tcl \
       tcl/lib/ns-qsnode.tcl \
```

```
+      tcl/bgp/ns-bgp-node.tcl \
+      tcl/bgp/ns-bgp-peerentry.tcl \
+      tcl/bgp/ns-rtProtoBGP.tcl \
+      tcl/bgp/ns-tcpmaster.tcl \
       @V_NS_TCL_LIB_STL@

  $(GEN_DIR)ns_tcl.cc: $(NS_TCL_LIB)
diff -rc ns-2.33.orig/routing/route.cc ns-2.33/routing/route.cc
*** ns-2.33.orig/routing/route.cc 2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/routing/route.cc     2008-07-26 20:39:07.000000000 -0700
***************
*** 393,406 ****
--- 393,426 ----
  {
       check(src);
       check(dst);
+ // Merged by Will Hrudey
+         // Modified by Tony Feng. check if src and dst come from the same as.
+         // Note that index = nodeid +1.
+         Tcl& tcl = Tcl::instance();
+      tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",src-1);
+      int as_num_src = atoi(tcl.result());
+         tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",dst-
1);
+      int as_num_dst = atoi(tcl.result());
+         if (as_num_src == as_num_dst)
+         // End Tony Feng.
       adj_[INDEX(src, dst, size_)].cost = cost;
  }
  void RouteLogic::insert(int src, int dst, double cost, void* entry_)
  {
       check(src);
       check(dst);
+ // Merged by Will Hrudey
+         // Modified by Tony Feng. check if src and dst come from the same AS.
+         Tcl& tcl = Tcl::instance();
+      tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",src-1);
+      int as_num_src = atoi(tcl.result());
+         tcl.evalf("[[Simulator instance] get-node-by-id %d] set as_num_",dst-
1);
+      int as_num_dst = atoi(tcl.result());
+         if (as_num_src == as_num_dst) {
+         // End Tony Feng.
       adj_[INDEX(src, dst, size_)].cost = cost;
       adj_[INDEX(src, dst, size_)].entry = entry_;
+      }
  }

  void RouteLogic::reset(int src, int dst)
diff -rc ns-2.33.orig/routing/rtmodule.cc ns-2.33/routing/rtmodule.cc
*** ns-2.33.orig/routing/rtmodule.cc    2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/routing/rtmodule.cc  2008-07-26 20:39:16.000000000 -0700
***************
*** 142,147 ****
--- 142,158 ----
       bind("classifier_", (TclObject**)&classifier_);
  }

+ // Merged by Will Hrudey
+ // Added by Tony Feng for BGP.
+ static class BGPRoutingModuleClass : public TclClass {
+ public:
+      BGPRoutingModuleClass() : TclClass("RtModule/BGP") {}
```

```
+      TclObject* create(int, const char*const*) {
+             return ( new BGPRoutingModule );
+      }
+ } class_bgp_module;
+ // End Tony Feng.
+
  int RoutingModule::command(int argc, const char*const* argv)
  {
       Tcl& tcl = Tcl::instance();
***************
*** 509,511 ****
--- 520,546 ----
             next_rtm_->add_route(dst, target);
  }

+ // Merged by Will Hrudey
+ // Added by Tony Feng for BGP.
+ BGPRoutingModule::BGPRoutingModule() { }
+
+ int BGPRoutingModule::command(int argc, const char*const* argv) {
+      Tcl& tcl = Tcl::instance();
+      if ( argc == 3 ) {
+             if ( strcmp(argv[1], "route-notify") == 0 ) {
+               Node *node = (Node *)(TclObject::lookup(argv[2]));
+                    if (node == NULL) {
+                           tcl.add_errorf("Invalid node object %s", argv[2]);
+                           return TCL_ERROR;
+                    }
+                    if (node != n_) {
+                           tcl.add_errorf("Node object %s different from n_",
argv[2]);
+                           return TCL_ERROR;
+                    }
+                    n_->route_notify(this);
+                    return TCL_OK;
+             }
+      }
+      return (RoutingModule::command(argc, argv));
+ }
+ // End Tony Feng.
diff -rc ns-2.33.orig/routing/rtmodule.h ns-2.33/routing/rtmodule.h
*** ns-2.33.orig/routing/rtmodule.h     2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/routing/rtmodule.h   2008-07-26 20:39:21.000000000 -0700
***************
*** 67,72 ****
--- 67,77 ----
  class Node;
  class VirtualClassifier;
  class DestHashClassifier;
+ // Merged by Will Hrudey
+ // Added by Tony Feng for BGP.
+ class IPv4Classifier;
+ class rtProtoBGP;
+ // End Tony Feng.


  class RoutingModule : public TclObject {
***************
*** 178,181 ****
--- 183,198 ----
       virtual void add_route(char *dst, NsObject *target){}
  };
```

```
+ // Merged by Will Hrudey
+ // Added by Tony Feng for BGP.
+ class BGPRoutingModule : public RoutingModule {
+ public:
+         BGPRoutingModule();
+         virtual const char* module_name() const {return "BGP";}
+         virtual int command (int argc, const char* const * argv);
+ protected:
+         IPv4Classifier *classifier_;
+         rtProtoBGP *bgp_agent_;
+ };
+ // End Tony Feng.
  #endif //  ns_rtmodule_h
diff -rc ns-2.33.orig/tcl/lib/ns-default.tcl ns-2.33/tcl/lib/ns-default.tcl
*** ns-2.33.orig/tcl/lib/ns-default.tcl 2008-03-31 19:00:23.000000000 -0700
--- ns-2.33/tcl/lib/ns-default.tcl      2008-07-26 20:39:50.000000000 -0700
***************
*** 1344,1349 ****
--- 1344,1368 ----
  Agent/rtProto/DV set INFINITY          [Agent set ttl_]
  Agent/rtProto/DV set advertInterval     2

+ # Merged by Will Hrudey
+ # Added by Tony Feng for BGP
+ Agent/rtProto/BGP set connretry_interval_ 120
+ Agent/rtProto/BGP set masoi_ 15
+ Agent/rtProto/BGP set cluster_num 0
+ Agent/rtProto/BGP set bgp_id_ 0
+ Agent/rtProto/BGP set as_num_ 0
+ Agent/rtProto/BGP set auto_config_ false
+ Agent/rtProto/BGP set preference_  80
+
+ # PeerEntry
+ Agent/PeerEntry set ipaddr_ 0
+ Agent/PeerEntry set as_num_ 0
+ Agent/PeerEntry set bgp_id_ 0
+ Agent/PeerEntry set return_ipaddr_ 0
+ Agent/PeerEntry set hold_time_ 90
+ Agent/PeerEntry set keep_alive_interval_ 30
+ Agent/PeerEntry set mrai_ 30
+ # End Tony Feng
  Agent/Encapsulator set status_ 1
  Agent/Encapsulator set overhead_ 20

diff -rc ns-2.33.orig/tcl/lib/ns-lib.tcl ns-2.33/tcl/lib/ns-lib.tcl
*** ns-2.33.orig/tcl/lib/ns-lib.tcl     2008-03-31 19:00:23.000000000 -0700
--- ns-2.33/tcl/lib/ns-lib.tcl   2008-07-26 20:39:58.000000000 -0700
***************
*** 229,234 ****
--- 229,243 ----
  }

  source ns-qsnode.tcl
+ # Mergedy by Will Hrudey
+ # Added by Tony Feng for BGP
+ source ../bgp/ns-bgp-node.tcl
+ source ../bgp/ns-rtProtoBGP.tcl
+ source ../bgp/ns-bgp-peerentry.tcl
+
+ #TCPMaster
+ source ../bgp/ns-tcpmaster.tcl
+ # End Tony Feng
```

```
  # Obsolete modules
  #source ns-wireless-mip.tcl
***************
*** 395,400 ****
--- 404,421 ----
       }
  }

+ # Merged by Will Hrudey
+ # Added by Tony Feng for BGP
+ Simulator instproc BGP { val } {
+      if { $val == "ON" } {
+            Node enable-module BGP
+            Node disable-module Base
+      } else {
+            Node disable-module BGP
+            Node enable-module Base
+      }
+ }
+ # End Tony Feng

  Simulator instproc PGM { val } {
          if { $val == "ON" } {
***************
*** 1102,1107 ****
--- 1123,1136 ----
       # Register this simplex link in nam link list. Treat it as
       # a duplex link in nam
       $self register-nam-linkconfig $link_($sid:$did)
+ # Merged by Will Hrudey
+      # Added by Tony Feng for BGP.
+      if { [$n1 set as_num_] != [$n2 set as_num_] } {
+      # n1 and n2 reside in different AS, add a route to n2 in n1's classifier.
+            $n1 add-route [$n2 set address_] [[set link_($sid:$did)] set
head_]
+            $n1 cmd add-AS-neighbor $n2
+      }
+      # End Tony Feng.
  }

  #
diff -rc ns-2.33.orig/tcl/lib/ns-node.tcl ns-2.33/tcl/lib/ns-node.tcl
*** ns-2.33.orig/tcl/lib/ns-node.tcl    2008-03-31 19:00:23.000000000 -0700
--- ns-2.33/tcl/lib/ns-node.tcl  2008-07-26 20:40:09.000000000 -0700
***************
*** 66,82 ****
       eval $self next $args

          $self instvar id_ agents_ dmux_ neighbor_ rtsize_ address_ \
!                  nodetype_ multiPath_ ns_ rtnotif_ ptnotif_

       set ns_ [Simulator instance]
       set id_ [Node getid]
       $self nodeid $id_   ;# Propagate id_ into c++ space

       if {[llength $args] != 0} {
!            set address_ [lindex $args 0]
       } else {
            set address_ $id_
       }
       $self cmd addr $address_; # Propagate address_ into C++ space
       #$ns_ add-node $self $id_
       set neighbor_ ""
```

```
--- 66,96 ----
       eval $self next $args

          $self instvar id_ agents_ dmux_ neighbor_ rtsize_ address_ \
!                  nodetype_ multiPath_ ns_ rtnotif_ ptnotif_ as_num_

       set ns_ [Simulator instance]
       set id_ [Node getid]
       $self nodeid $id_   ;# Propagate id_ into c++ space

+ # Merged by Will Hrudey
+       # Modified by Tony Feng for BGP
       if {[llength $args] != 0} {
!                  if {[llength $args] == 1} {
!              set arg_0 [lindex $args 0]
!                  if { [scan $arg_0 "%s" ] != -1 } {
!                         # create node with arg_0 of [as_num:ipaddr] format
!                         $self parse-addr $arg_0
!                  } else {
!                         # create node with arg_0 of int value
!                         set address_ $arg_0
!                         set as_num_ 0
!                  }
!                  }
       } else {
              set address_ $id_
+             set as_num_ 0
       }
+      # End Tony Feng
       $self cmd addr $address_; # Propagate address_ into C++ space
       #$ns_ add-node $self $id_
       set neighbor_ ""
diff -rc ns-2.33.orig/tcp/rq.cc ns-2.33/tcp/rq.cc
*** ns-2.33.orig/tcp/rq.cc 2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/tcp/rq.cc      2008-07-26 20:40:44.000000000 -0700
***************
*** 298,305 ****
   * last seq# number in the segment plus one
   */

  TcpFlag
! ReassemblyQueue::add(TcpSeq start, TcpSeq end, TcpFlag tiflags, RqFlag
rqflags)
  {

       int needmerge = FALSE;
--- 298,306 ----
   * last seq# number in the segment plus one
   */

+ // Merged by Will Hrudey
  TcpFlag
! ReassemblyQueue::add(TcpSeq start, TcpSeq end, TcpFlag tiflags, RqFlag
rqflags, AppData* data)
  {

       int needmerge = FALSE;
***************
*** 328,333 ****
--- 329,336 ----
              head_->pflags_ = tiflags;
              head_->rqflags_ = rqflags;
              head_->cnt_ = initcnt;
```

```
+ // Merged by Will Hrudey
+                 head_->data = data;                //Added by Zheng Wang for
BGP

              total_ = (end - start);

***************
*** 500,505 ****
--- 503,511 ----
              n->prev_ = p;
              n->next_ = q;

+ // Merged by Will Hrudey
+                 n->data = data; //Added by Zheng Wang for BGP
+
              push(n);

              if (p)
***************
*** 529,534 ****
--- 535,542 ----
              else if (rcv_nxt_ >= start) {
                     rcv_nxt_ = end;
              }
+ // Merged by Will Hrudey
+               toReceiveQueue->enqueue((TcpData*)(q->data)); //Added by Zheng
Wang for BGP

              return tiflags;
       }
diff -rc ns-2.33.orig/tcp/rq.h ns-2.33/tcp/rq.h
*** ns-2.33.orig/tcp/rq.h  2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/tcp/rq.h       2008-07-26 20:40:56.000000000 -0700
***************
*** 68,73 ****
--- 68,75 ----

  #include <stdio.h>
  #include <stdlib.h>
+ // Merged by Will Hrudey
+ #include "receive_queue.h" //Added by Zheng Wang for BGP

  /*
   * ReassemblyQueue: keeps both a stack and linked list of segments
***************
*** 99,111 ****
              TcpFlag      pflags_;     // flags derived from tcp hdr
              RqFlag rqflags_;    // book-keeping flags
              int    cnt_;        // refs to this block
       };

  public:
       ReassemblyQueue(TcpSeq& rcvnxt) :
              head_(NULL), tail_(NULL), top_(NULL), bottom_(NULL), hint_(NULL),
total_(0), rcv_nxt_(rcvnxt) { };
       int empty() { return (head_ == NULL); }
!      int add(TcpSeq sseq, TcpSeq eseq, TcpFlag pflags, RqFlag rqflags = 0);
       int maxseq() { return (tail_ ? (tail_->endseq_) : -1); }
       int minseq() { return (head_ ? (head_->startseq_) : -1); }
       int total() { return total_; }
--- 101,116 ----
              TcpFlag      pflags_;     // flags derived from tcp hdr
              RqFlag rqflags_;    // book-keeping flags
```

```
             int    cnt_;         // refs to this block
+ // Merged by Will Hrudey
+                 AppData* data;          // Added by Zheng Wang for BGP
       };

  public:
       ReassemblyQueue(TcpSeq& rcvnxt) :
             head_(NULL), tail_(NULL), top_(NULL), bottom_(NULL), hint_(NULL),
total_(0), rcv_nxt_(rcvnxt) { };
       int empty() { return (head_ == NULL); }
! // Merged by Will Hrudey
!      int add(TcpSeq sseq, TcpSeq eseq, TcpFlag pflags, RqFlag rqflags = 0,
AppData* data = 0); //Modified by Zheng Wang for BGP
       int maxseq() { return (tail_ ? (tail_->endseq_) : -1); }
       int minseq() { return (head_ ? (head_->startseq_) : -1); }
       int total() { return total_; }
***************
*** 121,126 ****
--- 126,133 ----
           return (clearto(rcv_nxt_));
       }
       void dumplist();    // for debugging
+ // Merged by Will Hrudey
+      void connRevQueue(ReceiveQueue* revQueue){toReceiveQueue = revQueue;}
//Added by Zheng Wang for BGP

       // cache of allocated seginfo blocks
       static seginfo* newseginfo();
***************
*** 143,148 ****
--- 150,158 ----
       // within TCP to set rcv_nxt and thus to set the ACK field.  It is also
       // used in the SACK sender as sack_min_

+ // Merged by Will Hrudey
+         ReceiveQueue* toReceiveQueue;  //Added by Zheng Wang for BGP
+
       TcpSeq& rcv_nxt_;   // start seq of next expected thing
       TcpFlag coalesce(seginfo*, seginfo*, seginfo*);
       void fremove(seginfo*);    // remove from FIFO
diff -rc ns-2.33.orig/tcp/scoreboard-rq.cc ns-2.33/tcp/scoreboard-rq.cc
*** ns-2.33.orig/tcp/scoreboard-rq.cc   2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/tcp/scoreboard-rq.cc 2008-07-26 20:41:24.000000000 -0700
***************
*** 84,90 ****

       for(int i = 0 ; i < tcph->sa_length() ; i++){
             //printf("l: %i r: %i\n", tcph->sa_left(i), tcph->sa_right(i));
!            rq_.add(tcph->sa_left(i), tcph->sa_right(i), 0);
       }
       changed_ = changed_  || (old_total != rq_.total());
       return 0;
--- 84,91 ----

       for(int i = 0 ; i < tcph->sa_length() ; i++){
             //printf("l: %i r: %i\n", tcph->sa_left(i), tcph->sa_right(i));
! // Merged by Will Hrudey
!            rq_.add(tcph->sa_left(i), tcph->sa_right(i), 0, NULL);
       }
       changed_ = changed_  || (old_total != rq_.total());
       return 0;
diff -rc ns-2.33.orig/tcp/tcp-full.cc ns-2.33/tcp/tcp-full.cc
*** ns-2.33.orig/tcp/tcp-full.cc 2008-03-31 19:00:28.000000000 -0700
```

```
--- ns-2.33/tcp/tcp-full.cc      2008-07-26 20:41:33.000000000 -0700
***************
*** 341,346 ****
--- 341,368 ----
       return;
  }

+ // Merged by Will Hrudey
+   /*
+    * send a string of nBytes, added by Zheng Wang for BGP
+    */
+ void
+ FullTcpAgent::advance_bytes(int nBytes, const char* const data, Continuation*
caller)
+ {
+      if (writeCont != NULL){
+        printf("write error - socket already in blocking write\n");
+        if (caller != NULL)
+          caller->failure();
+        return;
+      }
+      if(toSendQueue==NULL)  {
+        toSendQueue = new SendQueue();
+      }
+      if (data!=NULL) {
+        writeCont = caller;
+        toSendQueue->enqueue(nBytes,data);
+      }
+      advance_bytes(nBytes);
+ }
  /*
   * the byte-oriented interface: advance_bytes(int nbytes)
   */
***************
*** 464,473 ****
--- 486,501 ----
       case TCPS_LISTEN:
              cancel_timers();
              newstate(TCPS_CLOSED);
+ // Merged by Will Hrudey
+                 if(mySocket)        // added by Tony Feng, informs the socket
that tcp closed successfully
+                     mySocket->disconnected();
              finish();
              break;
       case TCPS_SYN_SENT:
              newstate(TCPS_CLOSED);
+ // Merged by Will Hrudey
+                 if(mySocket)        // added by Tony Feng, informs the socket
that tcp closed successfully
+                     mySocket->disconnected();
              /* fall through */
       case TCPS_LAST_ACK:
              flags_ |= TF_NEEDFIN;
***************
*** 715,720 ****
--- 743,751 ----
       int fillshole = (start == rcv_nxt_);
       int flags;

+ // Merged by Will Hrudey
+      AppData* data = pkt->userdata();   //Added by Zheng Wang for BGP
+
```

```
       // end contains the seq of the last byte of
       // in the packet plus one

***************
*** 723,730 ****
                        now(), name());
                abort();
        }
!
!       flags = rq_.add(start, end, tiflags, 0);

        //present:
        //
--- 754,761 ----
                        now(), name());
                abort();
        }
! // Merged by Will Hrudey
!       flags = rq_.add(start, end, tiflags, 0, data); //Modified by Zheng Wang
for BGP

        //present:
        //
***************
*** 748,753 ****
--- 779,828 ----

           return (flags);
  }
+ // Merged by Will Hrudey
+ //Added by Zheng Wang for BGP
+ void FullTcpAgent::read(char* buffer, int nbytes, Continuation* caller)
+ {
+       if (readCont != NULL){
+         printf("read error - socket already in blocking read\n");
+         if (caller != NULL)
+           caller->failure();
+         return;
+       }
+
+       // if requested data is in the buffer, get it from buffer
+       // We use the fact that a data object arrives in the first TCP segment.
+       if(nbytes <= dataReceived) {
+         dataReceived -= nbytes;
+         if(toReceiveQueue->is_empty()) {
+            printf("receive queue is empty, exit\n");
+           return;
+         }
+         toReceiveQueue->retrieve_data(nbytes,buffer);
+           caller->success();
+       } else {
+         inbuffer = buffer;
+         readCont = caller;
+         readSize = nbytes;
+       }
+  }
+
+ void FullTcpAgent::recvBytes(int bytes)
+  {
+       dataReceived+= bytes;
+       if((readCont != NULL) && (dataReceived >= readSize)) {
+           toReceiveQueue->retrieve_data(readSize,inbuffer);
+           mySocket->appCallWaiting = false;
```

```
+        //mySocket->app_call_waiting = NULL;
+        dataReceived -= readSize;
+        Continuation* rc = readCont;
+        readCont = NULL;
+        rc->success();
+
+      }
+      Agent::recvBytes(bytes);
+  }
+ // End Zheng Wang

  /*
   * utility function to set rcv_next_ during inital exchange of seq #s
***************
*** 907,912 ****
--- 982,999 ----
  //printf("%f(%s)[state:%s]: sending pkt ", now(), name(), statestr(state_));
  //prpkt(p);
  //}
+ // Merged by Will Hrudey
+      //Added by Zheng Wang for BGP
+             //Set data field
+             if(toSendQueue)
+             {
+                    if(!toSendQueue->is_empty())
+                    {
+                           TcpData* pData = toSendQueue-
>get_data(seqno,datalen);
+                           p->setdata(pData);
+                    }
+             }
+      //End Zheng Wang

       send(p, 0);

***************
*** 1283,1288 ****
--- 1370,1384 ----

       if (ackno == maxseq_) {
              cancel_rtx_timer(); // all data ACKd
+ // Merged by Will Hrudey
+              // Added by Tony Feng for Socket::wirte()
+                  if(writeCont != NULL) {
+                     Continuation *  app_call_waiting = writeCont;
+                     writeCont = NULL;
+                     mySocket->appCallWaiting = false;
+                     app_call_waiting->success();
+                  }
+                  // End Tony Feng
       } else if (progress) {
              set_rtx_timer();
       }
***************
*** 1623,1628 ****
--- 1719,1726 ----
                     //     changes DELACK to ACKNOW and calls tcp_output()
                     rcv_nxt_ += datalen;
                     flags_ |= TF_DELACK;
+ // Merged by Will Hrudey
+                    toReceiveQueue->enqueue((TcpData*)pkt->userdata()); //
Added by Zheng Wang for BGP
                     recvBytes(datalen); // notify application of "delivery"
```

```
                    //
                    // special code here to simulate the operation
***************
*** 1705,1711 ****
--- 1803,1819 ----
                    fid_ = iph->flowid();
               }

+ // Merged by Will Hrudey
+             //Modified by Zheng Wang
+             if( mySocket!=NULL && mySocket->isListening) {
+                         hdr_ip* iph1 = hdr_ip::access(pkt);
+                         //Received SYN for listening tcp, we create a new
reading socket.
+                      tcpMaster->newInComing(pkt,mySocket);
+                         return;
+             }  else {
              newstate(TCPS_SYN_RECEIVED);
+                  }
+             // End Zheng Wang
              goto trimthenstep6;

          /*
***************
*** 1793,1798 ****
--- 1901,1911 ----
                          flags_ &= ~TF_NEEDFIN;
                          tiflags &= ~TH_SYN;
                    } else {
+ // Merged by Will Hrudey
+                                 //Added by Zheng Wang
+                                 if (mySocket)
+                               mySocket->connected();
+                                 //End Zheng Wang
                          newstate(TCPS_ESTABLISHED);
                    }

***************
*** 2025,2030 ****
--- 2138,2145 ----
                    newstate(TCPS_FIN_WAIT_1);
                         flags_ &= ~TF_NEEDFIN;
                  } else {
+ // Merged by Will Hrudey
+                        mySocket->listeningSocket->addConnection(mySocket);
//Added by Zheng Wang for BGP
                         newstate(TCPS_ESTABLISHED);
                    }
              if (ecn_ && ect_ && ecn_syn_ && fh->ecnecho() )
***************
*** 2276,2281 ****
--- 2391,2402 ----
              case TCPS_CLOSING:  /* simultaneous active close */;
                    if (ourfinisacked) {
                          newstate(TCPS_CLOSED);
+ // Merged by Will Hrudey
+                                 // Added by Tony Feng for BGP
+                                 if(mySocket)  {
+                                      mySocket->disconnected(); // Informs the
socket that tcp closed successfully
+                                 }
+                                 // End Tony Feng
                          cancel_timers();
```

```
                    }
                    break;
***************
*** 2289,2294 ****
--- 2410,2421 ----
                    // K: added state change here
                    if (ourfinisacked) {
                          newstate(TCPS_CLOSED);
+ // Merged by Will Hrudey
+                                 // Added by Tony Feng for BGP
+                                 if(mySocket)  {
+                                      mySocket->disconnected(); // Informs the
socket that tcp closed successfully
+                                 }
+                                 // End Tony Feng
                          finish(); // cancels timers, erc
                          reset(); // for connection re-use (bug fix from ns-
users list)
                          goto drop;
***************
*** 2576,2582 ****
        * Due to F. Hernandez-Campos' fix in recv(), we may send an ACK
        * while in the CLOSED state.  -M. Weigle 7/24/01
        */
!      if (state_ == TCPS_LISTEN) {
             // shouldn't be getting timeouts here
                  if (debug_) {
                     fprintf(stderr, "%f: FullTcpAgent(%s): unexpected timeout
%d in state %s\n",
--- 2703,2710 ----
        * Due to F. Hernandez-Campos' fix in recv(), we may send an ACK
        * while in the CLOSED state.  -M. Weigle 7/24/01
        */
! // Merged by Will Hrudey
!      if (state_ == TCPS_LISTEN && (mySocket==NULL || !mySocket->isListening))
{ // Modified by Tony Feng for BGP
             // shouldn't be getting timeouts here
                  if (debug_) {
                     fprintf(stderr, "%f: FullTcpAgent(%s): unexpected timeout
%d in state %s\n",
diff -rc ns-2.33.orig/tcp/tcp-full.h ns-2.33/tcp/tcp-full.h
*** ns-2.33.orig/tcp/tcp-full.h  2008-03-31 19:00:28.000000000 -0700
--- ns-2.33/tcp/tcp-full.h 2008-07-26 20:41:40.000000000 -0700
***************
*** 39,44 ****
--- 39,52 ----

  #include "tcp.h"
  #include "rq.h"
+ // Merged by Will Hrudey
+ //Added by Zheng Wang
+ #include "send_queue.h"
+ #include "receive_queue.h"
+ #include "tcp_socket.h"
+ #include "tcp_master.h"
+ #include "continuation.h"
+ //End Zheng Wang

  /*
   * most of these defines are directly from
***************
*** 114,137 ****
  };
```

```
  class FullTcpAgent : public TcpAgent {
  public:
       FullTcpAgent() :
              closed_(0), pipe_(-1), rtxbytes_(0), fastrecov_(FALSE),
              last_send_time_(-1.0), infinite_send_(FALSE), irs_(-1),
              delack_timer_(this), flags_(0),
              state_(TCPS_CLOSED), recent_ce_(FALSE),
!             last_state_(TCPS_CLOSED), rq_(rcv_nxt_), last_ack_sent_(-1) { }
!
!      ~FullTcpAgent() { cancel_timers(); rq_.clear(); }
       virtual void recv(Packet *pkt, Handler*);
       virtual void timeout(int tno);    // tcp_timers() in real code
       virtual void close() { usrclosed(); }
       void advanceby(int);       // over-rides tcp base version
       void advance_bytes(int);   // unique to full-tcp
          virtual void sendmsg(int nbytes, const char *flags = 0);
          virtual int& size() { return maxseg_; } //FullTcp uses maxseg_ for
size_
       virtual int command(int argc, const char*const* argv);
              virtual void reset();                    // reset to a known point
  protected:
       virtual void delay_bind_init_all();
       virtual int delay_bind_dispatch(const char *varName, const char
*localName, TclObject *tracer);
--- 122,175 ----
  };

  class FullTcpAgent : public TcpAgent {
+ // Merged by Will Hrudey
+ friend class TcpMaster;
  public:
       FullTcpAgent() :
              closed_(0), pipe_(-1), rtxbytes_(0), fastrecov_(FALSE),
              last_send_time_(-1.0), infinite_send_(FALSE), irs_(-1),
              delack_timer_(this), flags_(0),
              state_(TCPS_CLOSED), recent_ce_(FALSE),
! // Merged by Will Hrudey
!             last_state_(TCPS_CLOSED), rq_(rcv_nxt_), last_ack_sent_(-1)
!                  {   // Modified by Zheng Wang for BGP
!                        toSendQueue = new SendQueue();
!                   toReceiveQueue = new ReceiveQueue();
!                        rq_.connRevQueue(toReceiveQueue);
!                   dataReceived = 0;
!                       readCont = NULL;
!                       writeCont = NULL;
!                     }
!
!      ~FullTcpAgent()
!                {    cancel_timers();
!                     rq_.clear();
!                     if(toSendQueue)
!                    delete toSendQueue;
!                if(toReceiveQueue)
!                    delete toReceiveQueue;
!                }    // End Zheng Wang
       virtual void recv(Packet *pkt, Handler*);
       virtual void timeout(int tno);    // tcp_timers() in real code
       virtual void close() { usrclosed(); }
       void advanceby(int);       // over-rides tcp base version
       void advance_bytes(int);   // unique to full-tcp
+ // Merged by Will Hrudey
+      // Added by Zheng Wang for BGP
```

```
+      void advance_bytes(int nBytes, const char* const data, Continuation*
caller);
+      void recvBytes(int bytes);  // Overrides Agent's recvBytes();
+      void read(char* buffer, int nbytes, Continuation* caller);
+      // End Zheng Wang
          virtual void sendmsg(int nbytes, const char *flags = 0);
          virtual int& size() { return maxseg_; } //FullTcp uses maxseg_ for
size_
       virtual int command(int argc, const char*const* argv);
             virtual void reset();                   // reset to a known point
+ // Merged by Will Hrudey
+      // Added by Zheng Wang
+      ReceiveQueue* getRevQueue() {return toReceiveQueue;}
+         TcpSocket* mySocket;
+         TcpMaster* tcpMaster;
+      // End Zheng Wang
  protected:
       virtual void delay_bind_init_all();
       virtual int delay_bind_dispatch(const char *varName, const char
*localName, TclObject *tracer);
***************
*** 238,243 ****
--- 276,292 ----
       int last_ack_sent_; /* ackno field from last segment we sent */
       double recent_;            // ts on SYN written by peer
       double recent_age_; // my time when recent_ was set
+ // Merged by Will Hrudey
+      // Added by Zheng Wang for BGP
+      SendQueue* toSendQueue;
+      ReceiveQueue* toReceiveQueue;
+      int dataReceived;
+      Continuation* readCont;
+         Continuation* writeCont;
+      int readSize;
+         int writeSize;
+      char* inbuffer;
+      // End Zheng Wang

       /*
        * setting iw, specific to tcp-full, called
```

11.7 Installation Steps

The following installation steps must be followed to install, compile, and run ns-2.33 with ns-BGP 2.0:

1. Download the ns-2.33 release from the ns-2 homepage: ns-allinone-2.33.tar.gz

2. Unpack the ns-2 software archive into your home directory:

   ```
   $ gzip -dc ns-allinone-2.33.tar.gz | ( cd ; tar xvf - )
   ```

   This will create a ns-allinone-2.33 directory within your home directory.

3. Run the ns-2 installation script and validate the installation:

   ```
   $ cd ~/ns-allinone-2.33 ; ./install
   ```

```
$ cd ns-2.33; ./validate  (optional)
```

Refer to the ns-2 installation README file and the website for further installation details if necessary.

4. Once the installation is complete, the following output is displayed, requesting the configuration of two environment variables. You may also optionally update your PATH variable:

IMPORTANT NOTICES:

(1) You MUST put /home/<user>/ns-allinone-2.33/otcl-1.13, /home/<user>/ns-allinone-2.33/lib,
into your LD_LIBRARY_PATH environment variable.
If it complains about X libraries, add path to your X libraries
into LD_LIBRARY_PATH.
If you are using csh, you can set it like:
setenv LD_LIBRARY_PATH <paths>
If you are using sh, you can set it like:
export LD_LIBRARY_PATH=<paths>

(2) You MUST put /home/<user>/ns-allinone-2.33/tcl8.4.18/library into your TCL_LIBRARY environmental variable. Otherwise ns/nam will complain during startup.

After these steps, you can now run the ns validation suite with
cd ns-2.33; ./validate

For trouble shooting, please first read ns problems page
http://www.isi.edu/nsnam/ns/ns-problems.html. Also search the ns mailing list archive
for related posts.

5. Unpack the ns-2.33-BGP software archive into the ns-allinone-2.33 directory.

```
$ gzip -dc ns-2.33-bgp_2.0.tgz | (cd ~/ns-allinone-2.33; tar xvf – )
```

This will create a patch file ns-2.33-bgp_2.0_patch in the ns-allinone-2.33 directory along with the core ns-bgp source sub-directories and files.

6. Apply the ns-BGP patch file:

```
$ cd ~/ns-allinone-2.33 ; patch –p0 < ns-2.33-bgp_2.0_patch
```

The command sequence will patch 16 files with the necessary integration logic.

```
$ patch -p0 < ns-2.33-bgp_2.0_patch
patching file ns-2.33/common/node.cc
patching file ns-2.33/common/node.h
patching file ns-2.33/common/packet.h
patching file ns-2.33/common/simulator.cc
```

```
patching file ns-2.33/Makefile.in
patching file ns-2.33/routing/route.cc
patching file ns-2.33/routing/rtmodule.cc
patching file ns-2.33/routing/rtmodule.h
patching file ns-2.33/tcl/lib/ns-default.tcl
patching file ns-2.33/tcl/lib/ns-lib.tcl
patching file ns-2.33/tcl/lib/ns-node.tcl
patching file ns-2.33/tcp/rq.cc
patching file ns-2.33/tcp/rq.h
patching file ns-2.33/tcp/scoreboard-rq.cc
patching file ns-2.33/tcp/tcp-full.cc
patching file ns-2.33/tcp/tcp-full.h
```

7. Compile the environment

```
$ cd ~/ns-allinone-2.33/ns-2.33
$ ./configure
$ make clean ; make
```

The compilation should complete successfully with the final entries as follows:

```
<…>
make[1]: Leaving directory `/home/<user>/nsTest/3/ns-allinone-
2.33/ns-2.33/indep-utils/webtrace-conv/epa'
make[1]: Entering directory `/home/<user>/nsTest/3/ns-allinone-
2.33/ns-2.33/indep-utils/webtrace-conv/nlanr'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/home/<user>/nsTest/3/ns-allinone-
2.33/ns-2.33/indep-utils/webtrace-conv/nlanr'
make[1]: Entering directory `/home/<user>/nsTest/3/ns-allinone-
2.33/ns-2.33/indep-utils/webtrace-conv/ucb'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/home/<user>/nsTest/3/ns-allinone-
2.33/ns-2.33/indep-utils/webtrace-conv/ucb'
```