

**Experimentation with Multi-threaded, Distributed
Routing Technology in the Internet**

Final Report

(NCR 9318902, NCR-9612764)

Craig Labovitz, Farnam Jahanian

Merit Network. Inc.
4251 Plymouth Rd. Suite 2000
Ann Arbor MI 48105-2785

The University of Michigan
Department of Electrical Engineering and Computer Science
1301 Beal Avenue
Ann Arbor MI 481095-21227

Contents

1. Project Summary	3
1.1 Benefits to the Community	3
1.2 Collaborations.....	4
2. MRT Architecture	5
3. Deliverables	6
3.1 Deployment of MRTd on Wide-area CAIRN ATM Network	6
3.2 Next Generation Routing Protocols.....	7
3.3 Development of Testing/Simulation Tools with RTCL	9
3.4 Methodologies for Benchmarking and Evaluating MRT	13
3.5 MRT Technology Transfer.....	14
4. Use of MRT by Other Projects	17
4.1 Prototyping and IPv6.....	18
4.2 Software Platforms	18
4.3 Journal Citations.....	18
5. Bibliography	20
5.1 Publications Resulting from this Project that Cite NCR-9612764.....	20
5.2 References	20

1. Project Summary

Under the National Science Foundation Grant, “Experimentation with Multi-threaded, Distributed Routing Technology in the Internet” (NCR-9612764), Merit Network, Inc., and The University of Michigan Department of Electrical Engineering and Computer Science (EECS) developed a scalable, distributed, multi-threaded architecture for the management, simulation, and distribution of inter-domain policy-based routing information for the Internet. Known as the Multi-Threaded Routing Toolkit (MRT), the pioneering toolkit supports IPv4/IPv6 BGP, DVMRP, RIP/RIPng, PIM-DM, and OSPF on a variety of hardware and software architectures, including NetStar/Ascend, SunOS, BSDI, Linux, FreeBSD, and Solaris.

The MRT software can read Cisco configuration files and offers an interactive telnet configuration and monitoring interface. The toolkit can also:

1. Serve as the backbone routing software for an IPv6 or IPv4 network connection.
2. Simultaneously handle tasks such as routing policy communication, routing policy calculation, and maintenance of a RIB, and distribute these tasks over multiple processors or multiple machines.
3. Generate and analyze route flap statistics.
4. Generate real-time graphical maps of Internet routing.
5. Capture a BGP peering session and monitor it in real time.
6. Record and replay sequences of events, such as routing failures.

A prototype of this tool was developed under the initial NSF Routing Technology grant (NCR 9318902). The early version of the daemon lacked support for multiple protocols and provided a primitive configuration interface. Merit and EECS subsequently advanced MRT development to provide a robust, configurable routing daemon that serves as the production IPv6 software for a substantial segment of the 6bone.

The final release of the MRT toolset enjoyed a widespread, enthusiastic response from users in the research and vendor communities, as well as from Internet service providers. The source for the widely used code now resides at SourceForge.net, <http://sourceforge.net/projects/mrt>, the world's largest Open Source development web site. Technical user support is provided by and for the MRT community through the widely read mailing list, mrt-support@merit.edu.

1.1 Benefits to the Community

The MRT project served a diverse user community. The ISP community benefitted from the projects’ monitoring tools and routing statistics, which are still distributed to hundreds of operators daily through the routing-problems@merit.edu email list. The vendor community benefitted from the exploration of parallelism and multi-processor

technology used in conjunction with routing protocols. Several router vendors and NextHop Technologies (formerly the GateD Consortium) have adopted elements of the technology for use in their regular production cycles.

The research community benefitted from the parallelism and multi-processor routing technology research, and contributed to the knowledge base on the applications of parallelism, protocol simulation/testing, and the analysis of Internet dynamics. The research also supplied the research/academic communities with libraries to facilitate the rapid development and prototyping of routing protocols and policies. A number of research organizations adopted MRT technology for use in their research (see Section 4 below).

All MRT users have access to a complete set of documentation, including the *MRT Installation Guide*, the *MRT User/Configuration Guide*, and the *MRT Programmer's Guide*.

1.2 Collaborations

Merit and EECS worked with a number of partner organizations to validate the experimental MRT architecture. Under the previous Routing Technology grant, Merit fostered close relationships with network service providers and router vendors. Their continuing feedback influenced the overall direction of this project, and was instrumental in defining the robustness and performance requirements on the MRT architecture. Router vendors, including Cisco and Bay Networks, continued to follow the results of our research and subsequently adopted some of the resulting technology. As detailed below, partners also included the CAIRN backbone network and several software vendors.

In collaboration with the University of Michigan Real Time Computing Laboratory (RTCL), Merit developed graphical user interface tools based on MRT modules for the testing, simulation and analysis of commercial and experimental routing protocol implementations. See Section 3.3 for further information.

As noted in Section 3.2.1, much of the MRT project's work on IPv6 routing protocols was accomplished through collaborations with colleagues in Japan, Portugal, the Netherlands, the UK, and the US, who contributed bug fixes, patches, and Unix ports.

Throughout the project, Merit encouraged the transfer of MRT technology to commercial routing vendors by technology demonstrations, dissemination of research results through the web, technical conferences, and the public release of software tools. Several router vendors adopted MRT simulation tools for regular use in their product development cycles. Network service providers, including Sprint, MCI and ANS, actively used MRT-based tools to monitor their networks and debug routing problems.

2. MRT Architecture

The MRT project explored a number of novel approaches to routing architecture design and protocol implementation. Research topics included methods of parallelizing routing protocol computation, an evaluation of an object-oriented, library-based approach to routing protocol implementation, and the inclusion of recent advances in operating system technology into routing protocol implementations. The MRT tools make use of parallelized light-weight processes, multiple processor support, and shared memory to gain improvements in speed, scaling and robustness over traditional routing architectures. The object-oriented, library-based modular design of MRT encourages the rapid addition and prototyping of experimental routing protocol and inter-domain routing policy algorithms.

The growth in the size and topological complexity of the Internet has placed severe strains on the backbone routing infrastructure. Ongoing network problems, including route flaps, limits on routing table memory, protocol implementation bugs, and configuration errors have led to failures of large segments of the Internet on a regular basis. The frequency and severity of these network outages will only grow as the Internet continues to expand.

The traditional model of routing architecture, where packet forwarding, policy computation, and protocol communication all take place on a single platform, does not scale. Cost and technology constraints will eventually limit the ability of router vendors to add ever-faster processors and greater memory capacity to their products.

Almost all modern routing platforms can be modeled as a single, monolithic process that handles both routing protocol events and policy calculations. In these architectures, the forwarding of packets and the distribution of network reachability information are tightly coupled. This model is well suited for routing software that operates on a limited, uniprocessor machine and manages a small, finite quantity of routing information and policy calculations.

The monolithic architecture model, however, exhibits poor utilization of multi-processor machines. As the trend towards hardware vendors releasing multi-processor architectures continues (e.g., SGI Challenge, SPARCCenter 2000, Dec AlphaServer), a routing architecture is needed that could distribute operations over multiple processors as well as over multiple, distributed machines.

Internet researchers also had a growing need for a software platform on which to experiment and develop the next generation of routing and network policy protocols. The monolithic architecture of the routing development software platforms available at the beginning of the project, namely GateD, tended to greatly complicate and discourage the addition of new policy languages and routing protocols.

The MRT distributed, multi-threaded, object-oriented routing architecture is designed to meet the future needs of scalable backbone routing. Instead of a single thread of control, MRT can simultaneously handle a complex variety of inter-related tasks, including routing policy communication, routing policy calculation, and maintenance of a routing information base. MRT can distribute these tasks over multiple processors, as well as over multiple machines.

As an open, general purpose routing platform that runs in a Unix workstation environment, MRT pioneered the technology for high-speed routing and managing the exponential growth of Internet routing table paths.

3. Deliverables

The five major tasks associated with the MRT project are detailed below:

3.1 Deployment of MRTd on Wide-area CAIRN ATM Network

This goal was accomplished in Year 2 of the project, when CAIRN (<http://www.isi.edu/CAIRN/>), the successor to the nationwide DARTNet ATM network, deployed MRT software as the engine for providing real-time, dynamic routing information to its previously statically routed specialized ATM switches. The early 1999 deployment of MRT on the CAIRN project's routers provided an environment for evaluating the throughput, robustness, and adaptability of the multi-threaded, distributed MRT architecture. The CAIRN/MRT partnership also promoted collaboration between NSF and ARPA research activities.

CAIRN chose to deploy the NetStar/Ascend router in its backbone infrastructure. The NetStar architecture used inexpensive PCs to provide routing table information to dedicated switching hardware directly connected via an attached Ethernet.

In cooperation with CAIRN, Merit developed MRT software to run on the NetStar hardware and provide dynamic routing information. Based on CAIRN research goals, the MRT software incorporated support for IPv6, inter- and intra-domain routing policy, multicast, and bandwidth reservation protocols. In the CAIRN testbed, the NetStar architecture provided a powerful and flexible environment for experimentation with operating system and software routing technology. NetStar/Ascend subsequently adopted several of the MRT testing and simulation tools as part of its regular product development cycle.

The CAIRN project and Merit also used MRT as the routing software base for joint research into advanced new routing protocols and policy languages. As part of this research, Merit and CAIRN researched and implemented a number of the next-generation of routing protocols, including DVMRP, as well as IPv6-capable routing implementations. The collaboration with CAIRN allowed an evaluation how well the object-oriented MRT architecture met its software engineering goals of code-reuse and supporting the rapid development of new routing protocols. The work with CAIRN also

served as a “proof of concept” for MRT’s parallel-processing techniques, and made it possible to explore the benefits of the technology within the framework of an operational network environment.

3.2 Next Generation Routing Protocols

The need for parallel processing for dealing with routing protocols played an important role in MRT research. In 1997, at the beginning of the project, most routing policies and protocols were relatively simple. Consumer demand, however, rapidly drove ISPs to adopt far more complex policies and routing technology, such as BGP communities and DPA (Destination Preference Attribute). In light of the continuing advances in Routing Policy Specification Language (RPSL), QoS routing, multicast, and other technologies, there existed the potential for router CPU processing capacity to become the bottleneck in future Internet development. Parallelism provided a relatively inexpensive, scalable solution to meet these future processing needs.

During the first year of the project, Merit and EECS made substantial progress toward the development of software and infrastructure for the analysis of current and next-generation routing technology. Specifically, first-year project accomplishments included:

1. Release of an initial IPv4/IPv6 routing daemon and tool suite, which came into active use providing backbone routing for the 6bone. Data collected from IPv6 analysis tools uncovered a number of flaws in vendor IPv6 implementation.
2. Continued analysis and development of inter-domain routing tools. We uncovered a number of significant pathologies in inter-domain BGP Internet routing (see C. Labovitz, G. Robert Malan, Farnam Jahanian, “Internet Routing Instability,” *Proceedings of SIGCOMM'97* and *IEEE/ACM Transactions on Networking*, 6(5):515-528, October 1998.)
3. Initial development of an active/passive network analysis engine (see Inference Engine,” paper accepted for SIGCOMM '98. – what’s this?)
4. Initial development of intra-domain (OSPF) analysis tools, extending analysis techniques used on BGP in the inter-domain to OSPF in the intra-domain.

Second-year accomplishments focused on exploring routing software architecture design and technologies for the analysis and evaluation of current and emerging routing protocols and algorithms. Completed tasks included:

1. Development and deployment of software for the study of network inter-domain routing failures. This work led to significant findings involving the rate of failure and repair of large sections of the default-free core Internet. (See C. Labovitz, Abha Ahuja and Farnam Jahanian, “Experimental Study of Internet Stability and Backbone Failures.” *Proc. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing*, Madison, WI, June 15, 1999.)
2. Release of stable IPv6 inter and intra-domain routing protocols in the MRTd routing daemon. As noted, MRTd software now served as the primary routing

base in several network testbeds, including the CAIRN backbone and large segments of the 6bone.

3. MRTd daemon support for most of the rich policy and syntax language provided on commercial router platforms.
4. Development of a routing protocol traffic generator and simulator. Unlike pure simulators (e.g., NS), BGPsim combines recordings of “real-world” routing traffic with policy, and simulated traffic. The BGPsim tool is used extensively by both research groups and vendors to study the performance of routing products and protocols under varying loads. Internally, Merit also completed a major study of Internet routing convergence and fault-injection using the BGPsim tool.
5. Merit and EECS worked closely with major Internet router vendors and Internet providers on analysis of routing protocols and Internet performance. At least one router vendor adopted the multi-threaded protocol architecture pioneered during this project.

Accomplishments in the third year of the project and the one-year extension included completion of:

1. Intra-and inter-domain routing engines and analysis tools.
2. Fault-injection and traffic simulation/generation tool.
3. Evaluation of our multi-threaded routing protocol design.
4. Performance analysis/comparison with our research and commercial routing protocol engines.

3.2.1 MRT, IPv6 and the 6bone

The MRT software was uniquely positioned to explore routing protocol technologies for IPv6. IPv6 routing support was added to MRT by Masaki Hirabaru, an Associate Professor of Information Science at Japan's Nara Institute of Science and Technology, who worked at Merit during the 1996-1998 academic years. Hirabaru developed a wide-area IPv6 testing environment for MRT using the 6bone, a virtual network that links sites in North America, Europe, and Japan. The 6bone is layered on top of portions of the physical, IPv4-based Internet to support routing of IPv6 packets, as that function has not yet been integrated into all production routers.

Merit participated in national and international IPv6 research activities, including supporting tunnels and routing to the 6bone. Merit also coordinated research activities and IPv6 software development with other 6bone participants, including WIDE, the Widely Integrated Distributed Environment in Japan (<http://www.wide.ad.jp/>) and INRIA, a French national research organization (<http://www.inria.fr/index.en.html>). Gifts from Microsoft supported the development of MRTd on the experimental/research Microsoft NT IPv6 platform.

Support for RIPng and the IPv6 policy language was also implemented in MRT, including implementation of IPv6 access-lists and router descriptions. Merit also worked

with IPv6 operating system development groups to fix several bugs in the IPv6 kernel software.

By the middle of the project's second year, MRT provided IPv6 support for:

1. Solaris IPv6 Release 5 (Solaris 2.5 and 2.5.1)
2. Linux IPv6 (2.1.24 and later)
3. INRIA IPv6 (FreeBSD 2.1 and 2.1.5)

MRT's RIPng implementation were now interoperating with DEC, Cisco, and Bay Networks routers over the 6bone. In addition, MRT's BGP implementation had been extended to support the COMMUNITIES attribute. Much of the work towards these goals was accomplished through collaborations with colleagues in Japan, Portugal, the Netherlands, the UK, and the US, who contributed bug fixes, patches, and Unix ports.

Although significant research had been directed towards the theoretical study of routing protocols, few researchers had examined the operational behavior of these protocols once they were deployed in wide area networks. The process of implementing, deploying and experimenting with IPv6 and other next-generation routing protocols allowed Merit and EECS to study their functional and performance characteristics. In addition, working with CAIRN made it possible to channel the knowledge and experiences gained with IPv6 and other evolving protocols back into the research and standards communities.

3.3 Development of Testing/Simulation Tools with RTCL

The ability to rapidly prototype new protocol languages and routing policies is critical to the evolution of the Internet technology. As the pace of routing innovation quickened in the late '90's, early implementations provided an invaluable guide to theoretical limitations and potential implementation hurdles. Examples of emerging technologies included IPv6, multicast, BGP communities, and RPSL.

In collaboration with the EECS Real-Time Computing Laboratory (RTCL), Merit developed tools and graphical user interfaces for the simulation, testing and analysis of routing protocol implementations. The RTCL group had previously developed tools for testing IPv4 communication protocols. Merit and the RTCL applied this research towards the testing and evaluation of transport and routing protocols, including IPv6 protocol implementations.

The MRT/RTCL simulation tools fall into four categories:

1. Recording and replaying of network routing traffic
2. Synthetic workload generation
3. Orchestrate protocol correctness
4. Characterization of routing information

Each category is described further below.

3.3.1 Recording and Replaying of Network Routing Traffic

These tools provided mechanisms for the logging and replaying of all protocol traffic seen in any routing exchange, including all of the BGP traffic exchanged at a Network Access Point (NAP). The tools read log files of recorded routing packet events to build and inject copies of the routing packets directly onto a network media. The tools supported the replaying of packets at the same, real-time rate as they were recorded.

These simulation tools provided a deterministic, reproducible means of evaluating routing software and hardware performance under the identical routing traffic loads. The tools provided router vendors and Internet service with the ability to exactly reproduce “real-world” routing problems and scenarios. For example, the toolkit could be used by ISPs that wanted to evaluate how well a new router handled the volume of BGP routing updates exchanged at MAE-East, or by a researcher who wanted to repeatedly replay AADS BGP NAP data to evaluate the performance of new route flap dampening algorithms.

3.3.2 Synthetic Workload Generation

This category of MRT simulation tools provided a means of generating synthetic routing protocol traffic based on some textual or graphical abstract description language. The simulation tools allowed the specification of a variety of network characteristics, including the number of routers, the topological connectivity (autonomous system paths), and path selection. By describing various router parameters, new protocols could be evaluated under a variety of network environments.

For example, the following Cisco-like language could be used to describe the generation of BGP traffic flapping between two different links every 30 seconds:

```
network-list aspath-1 690 78 34 89
network-list aspath-2 690 56 34 89

router bgp 185
neighbor 198.108.60.244 remote-as 65
neighbor 198.108.60.244 flap aspath-1 and aspath-2 30 seconds
```

Lacking automated simulation tools such as described above, it is difficult, if not impossible to recreate “real-world” routing conditions using a local testbed. Even a large testbed (100+ routers) cannot generate the volume of routing instability and topological diversity currently seen at the Internet core. The MRT tools provided a means of generating traffic conditions exceeding worst-case scenarios seen to date at the Internet core. For example, MRT had the ability to generate instability exceeding 100+ BGP prefixes/second, routing tables with 100,000+ prefixes, and corrupted or non-compliant routing packets. Using such tools, protocol and network developers were able to explore the scaling properties of routing protocols and routers.

The tools allowed the description of various router, link, and routing path instability and failure parameters. Examples of instability parameters may include the mean time

between link failures, and the frequency of oscillation between the selection of alternate routing paths. The research included efforts to better characterize the flow of inter and intra-domain Internet routing information, and to develop appropriate simulation models for analyzing protocols and network technology.

The toolkit also allowed the hand-editing and specification of individual routing packets. For example, the following packet could be used test how a vendor implementation handles invalid RIPng packets, as when the metric is set above 16:

```
packet:  RIPng packet
type:    response
version: 1
ff::01 18
```

All of the MRT simulation tools included a facility for recording, editing and later replaying a log of network routing activity.

3.3.3 Orchestrate State

RTCL developed a suite of Tcl/Tk-based tools, called ORCHESTRA, that could be used to test communication protocols. The ORCHESTRA toolset allowed graphical description of control flow and state transitions of a target protocol. It also supported specification of scripts that could orchestrate a distributed computation into “hard-to-reach” states by ordering certain concurrent events. For example, by intercepting messages between participants in a protocol stack, the tool could delay, drop, reorder, duplicate, and modify messages. Furthermore, the script could introduce spontaneous (or unexpected) messages into the system to probe the participants and to orchestrate the system execution onto a particular path. This tool has been used to conduct extensive experimental studies of several commercial and prototype implementations of communication protocols including six vendor implementations of TCP/IP.

Under the NSF grant, Merit and EECS enhanced the ORCHESTRA toolset to allow the specification of routing protocols. Initially, the tools supported the specification of routing protocol state transitions that may be reached through the reception of external routing packets (e.g., BGP OpenSent to Established with the receipt of an Open). Furthermore, a protocol designer will be able to specify scripts which orchestrate a computation of a routing protocol into a particular state to test various timing and functional behavior. Later versions of the tools included software for instrumenting Unix kernel routing tables and protocol implementations to allow manipulation of certain internal state transitions in a routing protocol implementation which may be impossible to exercise via external events. Although the toolset had the capability for exhaustive testing of the state transitions as well as random testing of a protocol state space, the main objective of this tool was to assist a protocol designer in focusing on certain computation paths which are difficult to reach.

3.3.4 Characterization of Routing Information

Operational networks and testbeds, such as CAIRN and the 6bone, have an ongoing need for tools to characterize routing environments and detect potential routing problems. The MRT project developed a class of tools that provides network characterization and analysis by examining the logs of routing protocol packets recorded by the previously described MRT tools. Examples of the statistics and analysis provided by these tools includes: the origin and distribution of route flap in a network, a topological map of autonomous system connectivity, and the relative stability of network links.

Routing protocol analysis and network characterization tools play an invaluable role in the research and deployment of new routing technologies. The MRT tools described in this section were used to support the development of the CAIRN and 6bone testbeds. In addition, several commercial network service providers actively used prototypes of these MRT tools in their daily network operations. This joint effort exploits the past work at RTCL on testing and validation of communication protocols and distributed algorithms (see Dawson:94, 95a, 95b.) The tools developed at RTCL were used extensively to conduct experiments on the functional and timing behavior of various communication protocols. The tool suite allowed router vendors and network researchers to evaluate the compliance of their protocol implementations with current and emerging international standards.

The graphical user interface in this tool suite produced scripts that drove the MRT routing communication modules in establishing peering sessions and transmitting network reachability information. Test scripts included descriptions of routing protocol state-machine events, timing specifications for routing packet transmission, and stability and topological characteristics of a simulated network. Internet service providers and researchers used these tools to study and analyze complex IPv4/v6 routing environments, including networks with routing loops, asymmetric routing, and severe route flaps.

3.3.5 Libraries for the Prototyping of New Routing Protocols and Policies

A number of MRT libraries helped researchers rapidly develop new routing protocols and policies. The libraries include an interface to kernel routing tables, a user telnet interface, a Cisco configuration library, a patricia tree routing database, support for multiple event timers, and libraries for the construction of routing protocol packets. The MRT libraries also include a clearly defined application programming interface (API) and a manual with examples and tutorials on using the libraries. Following the traditional Unix model, researchers may select only the libraries they need and may interchange and create new libraries as required.

Route-ORCHESTRA. This is a GUI-based tool for specification and testing of routing protocols based on the ORCHESTRA toolset. The Route-ORCHESTRA tool can be used by a protocol designer to specify the state transitions of a routing protocol and to describe scripts that test the functional and timing behavior of the target protocol by ORCHESTRATING the computation into a particular path. We used the Route-

ORCHESTRA tool to conduct experiments with a number of routing protocols, including RIPng, bandwidth reservation, multicast, and policy routing.

RIPNGSim, BGPSim, OSPFSim, DVMRPSim. These tools provided the synthetic workload generation and routing traffic recording/replaying functions described previously. The tools support the logging and later replaying of routing protocol packets from disk and shared memory. The tools log all packets with time-stamps to provide for the later “real-time” replaying of network traffic. In addition, the simulation tools support the generation of synthetic routing traffic based on an abstract notation language that specifies network environments and routing protocol events. The description language may be created by hand, or specified using the Route-Orchestra tool. These simulation tools include support for the Monte Carlo simulation of network events and routing traffic, including the random generation of network prefixes and autonomous system paths, and the generation of routing path changes.

RouteIf. This analysis tool allows a user to simulate the effects of network topology and routing policy changes on an active network. The tool uses data from routing protocol traffic logged at one or more routing exchange points on a network to build a topological map and database of the network. The tool then uses this database to answer “what-if” questions on network changes. For example, an ISP may want to know how routing traffic will be affected if a BGP community list is altered.

RouteWatcher. This tool is a back-end routing protocol monitor and debugger that tracks all routing protocol events. The tool includes a means of dumping packets to disk, decoding packets, and maintaining a database of routing protocol states and routing table attributes (e.g., an OSPF link state database and the multiple BGP aspaths associated with a destination). The tool provides for the generation of reports describing traffic patterns and stability analysis, and includes an interface to RMON tools for the generation of alerts when routing traffic patterns exceed certain criteria (e.g. excessive levels of route flap).

3.4 Methodologies for Benchmarking and Evaluating MRT

Merit has developed tools and methodologies to benchmark and evaluate the MRT architecture. The use of parallel, light-weight processes provides speedup to route processing through the pipelining of route packet processing. Merit investigated the tradeoffs between the benefits of pipelining and the costs of synchronization between threads. Other performance issues requiring evaluation include the increased delays and memory usage requirements due to contention for critical routing resources, and the overhead of maintaining routing data in shared memory.

Merit used a threads-capable profiler to study the runtime performance of MRT and evaluate the relative costs of parallelism. Merit also developed simulation tools and experiments to compare and evaluate the relative speed, memory performance and throughput of MRT in relation to other publicly available routing software packages (routed, mrouterd, GateD).

Tools and methodologies were also developed to address many of the open research questions that arose during development of the MRT route server. These research topics included: how well the architecture scales beyond multiple processors to multiple machines, determination of the optimal model for thread organization and parallelization in route processing, and identification of the best model for memory management and garbage collection in light-weight thread environment.

The MRT development and benchmarking platforms were a four-processor SPARCstation 20 running Solaris IPv6 Release 5 on Solaris 2.5.1 with POSIX threads support, and Intel Pentium Pro and 486DX systems running INRIA IPv6 on FreeBSD 2.1.5 and Linux IPv6 2.1.24.

3.5 MRT Technology Transfer

The MRT project directly impacted the evolution of scalable and robust routing technology in the Internet. Throughout the project, Merit and EECS were active participants in national and international Internet working groups, conferences, and workshops. We provided presentations on MRT technology at meetings of the North American Network Operator's Group (NANOG), IETF, RIPE, SIGCOMM, CAIDA, IEEE INFOCOM, and other conferences.

Merit actively pursued the transfer of the MRT research to commercial routing vendors and network service providers by technology demonstrations and public release of software tools in multiple stages. MRT technology and the goals of the project continued to evolve through ongoing discussions with members of the network service provider community, including MCI, Sprint and ANS. The operational knowledge and experiences gained with MRT in the CAIRN collaboration directly benefited the participating members of the CAIRN project in developing scalable routing for high-speed networks.

3.5.1 Real Internet Consortium and MRT

In the third year of the project, RICD 1.0.0a was implemented in MRT. Developed by the Real Internet Consortium, <http://www.real-internet.org/index-e.html>, RICD is an HQLIP/SRSVP implementation that replaces QoS-based MOSPF and RSVP, respectively.

The goal of the RIC consortium is to develop a simple and integrated suite of Internet protocols with high speed QoS assurance and real multicast with a massively parallel router architecture. To date, the RIC has specified HQLIP (Hierarchical QoS Link Information Protocol), MRVP (Multiple Rendez-Vous Point multicast Protocol), SRSVP (Simple Resource ReserVation Protocol), SS (Service Specification), and SURL (Stream URL), as well additional documents on scheduling, CS (Comfortable Service) and the SM (Static Multicast) architecture.

It is widely recognized that, because of the speed limitations of electronic circuits, Tbps or Pbps routers can be constructed only with massively parallel architecture, i.e., with many elementary routers linked by a high speed backbone. On the other hand, it is

widely (and incorrectly) held that fine-grained QoS assurance of individual flows is impossible, because, on high speed Internet backbones, there are so many flows that signaling of the flows and routing (including policing and fair queuing) of packets of the flows does not scale.

It is not widely understood that the backbone scalability problems of QoS assurance is solved by the massively parallel nature of high speed routers. As a result, people in conventional standardization forums such as the ISO, ITU, and IETF are, in vain, trying to produce complex protocols to aggregate flows to reduce the complexity. However, it is easy to prove that such aggregation is impossible, at least with multicast or QoS-routed flows. Thus, in some IETF Working Groups, QoS assurance is finally given up and CoS (Cost of Service) assurance, which is not useful for most applications and often obfuscated by the fallacy of calling CoS QoS, is pursued.

Instead, we solve the scalability problem of real QoS assurance in a simple, straightforward fashion. With a properly architected, massively parallel router, the signaling and routing load can be distributed evenly over elementary routers. As flows reserve certain bandwidths, the number of flows in a link is at most the bandwidth of the link divided by the bandwidth of flows, and proportional to the bandwidth of the link. Thus, problems of scalability proportional to the number of flows are scalably solved by elementary routers, the number of which is proportional to the speed of the link or the number of flows in the link.

(Need help with underlined sentences).

As multicast routing table entries cannot be aggregated, multicast communication is a resource (routing table entry) reserving one. Moreover, bandwidth cannot be negotiated between receivers with different available bandwidth, and the bandwidth can only be determined by the sender, transport of which requires resource (bandwidth) reservation.

SRSVP combined with MRVP signals resource reservation requests (including bandwidth and routing table entries) for unicast and multicast communication. With resource reserved multicast, scalable flow control mechanisms to disable intruders consume the reserved resource, so it is essential that multicast protocols use an authorized root, such as a rendezvous point of a shared tree PIM. SRSVP is based on RSVP, but greatly simplified to support a PIM-like multicast model only, and to use TCP between adjacent routers for reliable state maintenance. MRVP specifies the communication protocol between multiple rendezvous points for fault tolerant multicast.

Many multicast routing protocols broadcast or flood information on individual multicast groups in a certain domain, expecting that interdomain multicast protocols can solve the scalability problem of flooding. However, flooding of information by the network makes the network unnecessarily intelligent, which is against the end-to-end principle of the Internet. According to the end-to-end principle, it is the end system's responsibility to obtain the necessary information on the multicast session. SM uses DNS to let the end systems and routers at the branches of multicast trees authoritatively know the locations of rendezvous points (RPs). By allowing administrators of individual multicast groups

to control the locations of RPs, the RPs can be located near the expected source locations. Thus, it is not necessary to use a source-centered tree, which is known to cause serious scalability problems. Other session information, such as required bandwidth, common both to unicast and multicast, should be known by end systems through some end-to-end mechanism such as a URL, and should be propagated to intermediate routers through a signaling protocol. Session announcements through a broadcast or multicast channel, such as that used by SAP, do not scale.

The remaining commonly seen misunderstandings are that there are still problems of QoS assurance with scalability problems. One example is that scheduling of QoS-assured flows had needed sorting and had temporal computational complexity proportional to the logarithm of the number of flows. This misunderstanding originated from ATM and was imported to the IETF Intserve model as a guaranteed service model. However, by carefully analyzing the nature of QoS assurance and Internet traffic (for example, a 100% delay guarantee is meaningless when 100% reachability is not or cannot be guaranteed), it is possible to construct a traffic model (CS, Comfortable Service) and scheduling algorithm (PPQ, Policed Priority Queuing) of QoS assurance with a complexity proportional to the number of flows. With CS, it is not meaningful that different receivers request different bandwidth, which is one of the reasons why SRSVP is simple.

The other example is that QoS routing was an NP (Non-deterministic Polynomial) difficult problem. While QoS assurance of both delay and jitter is NP hard, such assurance is not meaningful in the dynamic Internet, where rerouting may drastically change the current delay, in which case, the lower bound of delay can be given only by the distance between end points divided by the speed of light, which is invariant regardless of the network status. Thus, we are targeted to assure maximum delay (including jitter) only.

Finally, we consider policy issues. While policies of individual ISPs to select backbone paths are considered essential for the current best-effort, flat-rate Internet, the same logic is not applicable to QoS-assured communication over the Internet. When subscribers are charged proportionally to their use, which is the charging policy for QoS-assured communication, ISPs want to enclose their users and users' traffic within their AS to maximize their revenue, which violates anti-trust laws.

For QoS-assured communication, it is subscribers who control the policy. ISPs should set their identities and charging policy and subscribers should choose appropriate routes based on their policy. In most cases, subscriber policy will be as simple as “use the most inexpensive route satisfying the required QoS,” but subscribers may also specify explicit routes. Note that with multicast, only the sender side of the subscribers can have their own detailed policy, because receivers with conflicting policies cannot share a single tree. HQLIP is the routing protocol which announces link information such as available QoS and charge of the link with 256 levels of hierarchy. BGP or distance vector based routing, in which the best route is selected by the network according to the “best route” defined by the providers' policy, is unusable for QoS assured communication. HQLIP is designed with quick convergence of routes in mind to enable quick layer 3 restoration.

The RIC consortium has successfully developed a simple protocol suite of the next generation Internet for QoS assurance, multicast, session announcement, charging, routing, policy and layer 3 restoration, only because we attempted to solve all the interrelated problems at once by an integrated protocol suite, which makes it unnecessary to make some protocol overly generic. RIC is now in the final stages of polishing its protocols and reference implementations.

4. Use of MRT by Other Projects

The underlying MRT architecture represented a radical shift from the traditional approach to routing taken by most routing software implementations. Many network service providers came to rely on the MRT tools as the only publicly available source of real-time Internet routing statistics and topological information. Merit worked closely with network service providers to tailor these tools to meet the changing needs of network operators.

Research projects and companies worldwide adopted the MRT format for storing routing updates and other raw BGP data. See, for example, the RIPE NCC Routing Information Service (<http://www.ripe.net/ris/ris-index.html>).

The following email messages demonstrate the importance of MRT to the research, ISP, and vendor communities:

From: kc@nlanr.net Tue Feb 26 11:21:44 2002
Date: Fri, 20 Sep 1996 13:36:28 -0700 (PDT)
From: kc <kc@nlanr.net>
To: labovit@merit.edu

The MRT/Merit routing stability statistics have been *really critical* to my own ability to beat up ISPs into being better behaved routing citizens. The Netnow stuff is also a critical step toward consumers being able to objectively critically assess ISPs performance. Both of these were landmark steps in my opinion [and no ISP would have done them, so they're exactly what tax dollars are optimally used for --]

Such tools, and any MRT API/toolkit that could make their development/maintenance/upgrading easier, are really valuable to the community.

- kc

Return-path: mkb@netstar.com
Message-id: <199609181745.MAA26924@sun6.>
From: mkb@netstar.com (Kristy Brown)
To: labovit@merit.edu
Subject: paragraph on mrt
Cc: mkb@netstar.com

For the past several months, Netstar has been using the Multi-threaded Routing Tool in our continuing development of dynamic routing protocols. We consider this tool an essential part of our testing environment to simulate the large and complex routing traffic patterns of the Internet. The Merit staff have provided both guidance and insight into our use of this technology.

Kristy Brown
Senior Software Engineer
612-996-6882
mkb@netstar.com

4.1 Prototyping and IPv6

The MRT routing daemon was used to prototype and test experimental routing protocols by the National Institute of Standards (NIST) Switch Project, <http://snad.ncsl.nist.gov/itg/nistswitch/description/nistswitch.html>, and SRI International's Telecommunications and Distributed Processing Group (wireless), <http://www.darpa.mil/ito/research/netex/presentations/SastrySlides-SRI.pdf>

Projects recommending MRT for IPv6 routing and testing include:

1. The IPv6 testbed at Virginia Polytechnic Institute and State University, <http://www.visc.vt.edu/ipv6/>
2. The KAME project, <http://www.kame.net/project-overview.html>

4.2 Software Platforms

MRT is recommended for systems running Linux by:

1. MIT's Rpmfind.Net Server, http://rpmfind.net/linux/RPM/Networking_Admin.html
2. SuSE Linux, http://www.suse.com/index_us.html
3. The Tuxfinder software locator, <http://www.tuxfinder.com/>
4. The Debian Project, <http://www.debian.org/>
5. The *Linux Administrator's Security Guide*, <http://www.seifried.org/lasg/network/>

For a FreeBSD recommendation of MRT, see <http://people.freebsd.org/~picobsd/>,

4.3 Journal Citations

M. Kim and B. Noble, "SANE: stable agile network estimation." *Technical Report CSE-TR-432-00*, University of Michigan, Department of Electrical Engineering and Computer Science, August 2000.

- N. G. Duffield and Matthias Grossglauser, "Trajectory sampling for direct traffic observation", *Proceedings of SIGCOMM 2000*, pp. 271-282.
- Paul Francis, Sugih Jamin, Vern Paxson, Lixia Zhang, Daniel F. Gryniewicz, and Yixin Jin, "An Architecture for a Global Internet Host Distance Estimation Service." *Proc. IEEE INFOCOM 1999*, pp. 210-217.
- Timothy Griffin and Gordon T. Wilfong, "An Analysis of {BGP} Convergence Properties." *Proc. SIGCOMM '99*, pp. 277-288.
- D. Katabi, J. Wroclawski, "A framework for Scalable Global IP-Anycast (GIA)," MIT Laboratory for Computer Science, *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, July 2000.
- B. Krishnamurthy and J. Wang, "On network-aware clustering of web clients.: *Proc. SIGCOMM 2000*, pp. 97-110.
- Tony Mancill, "Linux In The Corporate Network?" *Business Communications Review*, Jan. 2000, <http://www.bcr.com/bcsmag/2000/01/p28.asp>
- P. Narvaez, *Routing Reconfiguration in IP Networks*. Doctoral thesis, MIT, May 2000.
- Paolo Narvaez, Kai-Yeung Siu, and Hong-Yi Tzeng, "New Dynamic {SPT} Algorithm Based on a Ball-and-String Model." *INFOCOM (2)*, 1999, pp. 973-981.
- V. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics." *Lawrence Livermore Lab Report LBNL-40319, UCB/CSD 97-945*, April 1997
- A. Shaikh, L. Kalampoukas, R. Dube, and A. Varma, "Routing Stability in Congested Networks: Experimentation and Analysis," *ACM SIGCOMM '00*, pages 163-174
- H. Uijterwaal, RIPE NCC, "Routing Information Service R.I.S." *RIPE-200, Design Document*, October 1999.
- Marcel Waldvogel, George Varghese, Jon Turner, Bernhard Plattner, "Scalable High-Speed Prefix Matching." *ACM Transactions on Computer Systems* 19(4), Nov. 2001.
- Xipeng Xiao and Lionel M. Ni, "Parallel Routing Table Computation for Scalable {IP} Routers." *Communication, Architecture, and Applications for Network-Based Parallel Computing*, 1998.

5. Bibliography

5.1 Publications Resulting from this Project that Cite NCR-9612764

- C. Labovitz, A. Ahuja, A. Bose, F. Jahanian, "An Experimental Study of Delayed Internet Routing Convergence." *Proc. ACM SIGCOMM '00*, pp. 175-187.
- C. Labovitz, Abha Ahuja and Farnam Jahanian, "Experimental Study of Internet Stability and Backbone Failures." *Proc. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing*, Madison, WI, June 15, 1999.
- C. Labovitz, G. Malan, and F. Jahanian. "Origins of Pathological Internet Routing Instability." *Proc. IEEE INFOCOM*, Mar. 1999.
- Lixin Gao and Jennifer Rexford, "Stable Internet routing without global coordination," in *Measurement and Modeling of Computer Systems*, pp. 307-317.
- C. Labovitz, G. Robert Malan, Farnam Jahanian, "Internet Routing Instability." *Proceedings of SIGCOMM'97 and IEEE/ACM Transactions on Networking*, 6(5):515-528, October 1998.
- C. Labovitz, A. Ahuja, R. Wattenhofer, S. Venkatachary, "The Impact of Policy and Topology on Delayed Internet Routing Convergence." *Proc. IEEE INFOCOM 2001*.

MRT Installation Guide (http://www.merit.edu/mrt/mrt_doc/)

MRT User/Configuration Guide (http://www.merit.edu/mrt/mrt_doc/)

MRT Programmer's Guide (http://www.merit.edu/mrt/mrt_doc/)

5.2 References

- S. Bradner, A. Mankin. *IPng Internet Protocol Next Generation*. Addison-Wesley, 1996.
- B. Catanzaro, *Multiprocessor System Architectures*,. Prentice Hall PTR, 1995.
- Classless Interdomain Routing Discussion Mailing List, ciderd-request@merit.edu
- B. Chinoy, "Dynamics of Internet Routing Information," *Proc. SIGCOMM '93*, pp. 45-52, September 1993.
- M. Cekleov et al, "SPARCCenter 2000: Multiprocessing for the 90's!" *Proc. IEE COMPCON*, pp. 345-353. San Francisco CA, February 1993.

- S. Dawson and F. Jahanian, "Deterministic Fault Injection of Distributed Systems," in *Lecture Notes in Computer Science: Unifying Theory and Practice of Distributed Computing*, Springer-Verlag, September 1994.
- S. Dawson and F. Jahanian, "Probing and fault injection of protocol implementations," in *Proc. Int. Conf. on Distributed Computer Systems*, pp. 351--359, Vancouver, B.C., May 1995.
- S. Dawson, F. Jahanian, T. Mitton, "A Software Fault Injection Tool on Real-Time Mach." *Proc. IEEE Real-Time Systems Symposium*, Dec. 1995
- D. Estrin, Y. Rekhter, S. Hotz, "Scalable Interdomain Routing Architecture." *Proc. SIGCOMM '92*, pp. 40-52. August 1992.
- M. Goldberg, G. Neufeld, M. Ito. "A Parallel Approach to OSI Connection-oriented Protocols," *Third IFIP International Workshop on Protocols for High-Speed Networks*, pages 219-232, May 1993.
- C. Huitema, *Routing in the Internet*, Prentice Hall PTR, 1995.
- S. Kleiman, D. Shah, B. Smalders. *Programming with Threads*. Prentice Hall PT, 1996.
- B. Lindgren, B. Krupczak, M. Ammar, K. Schwan, "An Architecture and Tool-kit for Parallel and Configurable Protocols," *Proc. International Conference on Network Protocols*. San Francisco, CA, October 1993.
- The POSIX Threads standard, IEEE Std. 1003.1c-1995.
- A. Mikler, J. Wong,, and V. Honavar, "An Object-Oriented Approach to Modeling and Simulation of Routing in Large Communication Networks." IASTATECS//TR95-09 (25), 1995.
- E. Nahum, D. Yates, J. Kurose, and D. Towsley, "Performance Issues in Parallelizing Network Protocols," *Proc. First Symposium on Operating System Design and Implementation*, Usenix Association, Nov 1994
- North American Network Operators (NANOG) mailing list,
<http://www.nanog.org/maillinglist.html>
- V. Paxson, "End-to-End Routing Behavior in the Internet." *IEEE/ACM Transactions on Networking* 5(5), pp. 601-615, 1997.
- J. Salehi, J. Kurose, D. Towsley, "The Effectiveness of Affinity-Based Scheduling in Multiprocessor Networking." *IEEE/ACM Transactions on Networking* 4(4), pp. 516-530,1996.

- D. Schmidt, R. Suda, "The Service Configurator Framework." *Proc. IEE Second International Workshop on Configurable Distributed Systems*, Pittsburgh, PA, March 1994.
- J. Yates, E. Nahum, J. Kurose, D. Towsley, "Networking Support for Large Scale Multiprocessor Servers." *Technical Report CMPSCI95-83* University of Massachusetts.
- Y. Rekhter and P. Gross, "Application of the Border Gateway Protocol 4(BGP-4)." RFC 1711, March 1995.