Increasing TCP Throughput with an Enhanced Internet Control Plane

Andy Bavier, Larry Peterson, Princeton University
Jack Brassil, Rick McGeer, David Reed, Puneet Sharma, Praveen Yalagandula, HP Laboratories
Alex Henderson, Larry Roberts, Anagran Inc.
Stephen Schwab, Roshan Thomas, Erik Wu, Sparta Inc. *Brian Mark*, George Mason University
Ben Zhao, UC Santa Barbara *Anthony Joseph*, UC Berkeley

***Abstract*— *CHART* seeks to improve the performance of operational DoD internets through the introduction of an intelligent network overlay. TCP performance – particularly between CONUS and forward-deployed components located in combat theaters – can be severely degraded due to high loss rates and long latencies. The lack of current information about network conditions in the core NIPRnet/SIPRnet further compounds the problem, because end hosts lack the data required to make intelligent routing decisions.**

Deploying *CHART's* enhanced control plane improves measurement and monitoring of unreliable communication links to provide current network state information to routers implemented in both software and hardware, enabling intelligent routing around faulty links. We describe the design of software and hardware routers sharing a common network 'sensing' infrastructure, the implementation of end-to-end Quality of Service via flow state aware routers, and a new network-aware TCP/IP stack for Linux end systems. Performance test results demonstrate that bulk file transfer throughput can be increased by as much as an order of magnitude in networks with severely impaired communication links.

I. Introduction

The *Control for High-Throughput Adaptive Resilient Transport* (CHART) [1] system provides enhanced network control plane functionality. The need for an intelligent control plane arises from the reality that Internet Protocols provide best effort service that makes no guarantee on the quality of routes, and end-to-end performance degrades rapidly in response to relatively minor loss of link quality. CHART implements a network-wide real-time network monitoring service that is reliable, efficient, scalable, and secure. This network state information enables an intelligent routing system to rapidly re-route around detected network faults. The CHART system's principal goal is to achieve a 10x improvement in the performance of bulk data transfer communicated via TCP/IP in the presence of a multiplicity of transient network impairments that might be present between CONUS and forward-deployed units.

CHART combines a novel adaptive routing infrastructure and a distributed network 'sensing' infrastructure to improve end-to-end performance across an unreliable network. The routing infrastructure has two complementary components sharing a common underlying network measurement and monitoring (i.e., sensing) system. The sensing infrastructure monitors the state of underlying network and conveys network state information to the routing infrastructure. The combined system adaptively routes around failed or congested links to maintain high end-to-end throughput.

The first component of the intelligent routing infrastructure is a software routing overlay based on *Chimera*, an updated and high-performance version of *Tapestry* [5]. The software overlay runs on a dispersed collection of commodity IA-32 computers running the Linux Fedora Core operating system. These computing nodes may be dedicated to supporting CHART, or may be a shared decentralized computing overlay such as *PlanetLab* [8]. Each software routing overlay node consists of a packet forwarding engine maintaining a list of primary and backup routes to destination networks. End systems (e.g., Windows-based personal computers) typically connect to an ingress overlay router via relatively low-bandwidth edge links (e.g., up to 100 Mbs). Selective access to the software overlay is enabled through use of the open-source *OpenVPN* [14] virtual private network secure tunneling software.

The second routing component uses new, strategically deployed high-performance Flow State Aware (FSA) routers on high-bandwidth links (e.g., 1-10 Gbs). These routers may be deployed either in the network core, or at the edge (i.e., network ingress or egress routers) where high-performance end systems require connectiv-

[1]This work was supported in part by DARPA Contract N66001-05-9-8904 (Internet Control Plane).

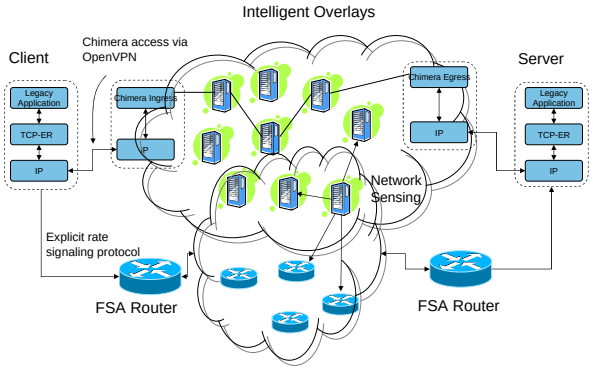

Fig. 1. CHART system architecture.

ity. Hardware routers monitor their outgoing links, detecting and re-routing around network failures in tens of milliseconds. Flow routers also differentiate between traffic classes, enhancing the end-to-end performance of high priority traffic. Primary and backup routes with link state information along each route is stored for every source-destination pair. When each packet is sent, its behavior is monitored to determine if its delivery is within the known Quality of Service (QoS) for that flow. The ability of flow routers to route separate flows to the same destination via separate paths increases the overall reliability of end-to-end communications. Figure 1 shows that the CHART architecture permits the co-existence and inter-operation of hardware and software routers; flow routers need not be present throughout the network.

A common decentralized network measurement and monitoring fabric communicates network state to routing components as well as adaptive end system applications. This sensing infrastructure securely aggregates and propagates measurements collected at both hardware and software monitors placed at a large number of network vantage points. Measurements collected include link transmission capacity (Mbs), packet loss rate, available bandwidth, path latency, and hop count [19].

CHART also implements several advanced traffic management and control features. An optional Linux-based explicit rate-aware protocols stack called *TCP-ER* enables clients of flow routers to accelerate data transmission by bypassing the 'slow-start' and 'congestion avoidance' phases of data transfer. In general, however, no system-level or application-level software modifications are required for an end-system to route packets through a CHART overlay.

The remainder of this document describes the CHART system and is organized as follows. Sections II and III describe software and hardware routing in detail. Section IV presents enhanced traffic management techniques, and performance test results are presented in Section V. The final section summarizes our overall approach to an enhanced control plane.

II. Software Routing

CHART uses a structured software routing overlay named *Chimera*, based on the *Tapestry* protocol [5]. Structured overlays conform to a specific graph structure that allows them to locate communication endpoints by exchanging $O(\log N)$ messages in an overlay of N nodes. A node represents an instance of a participant in the overlay (one or more nodes may be hosted by a single physical IP host). Participating nodes are assigned *nodeIDs* uniformly at random from a large identifier space. Application-specific objects are assigned unique identifiers called keys, selected from the same namespace. Chimera uses an identifier space of n-bit integers modulo $2n$ (n=160).

Chimera supports routing of messages with a given key to its root node (the message destination). Each node maintains a routing table of a small number of overlay neighbors. Messages are forwarded to neighbors whose nodeIDs are progressively closer to the key in the identifier space (*i.e.*, prefix routing). An important benefit of this routing approach is that any node satisfying the routing constraint can serve as a next routing hop. For example, in Chimera the first hop of a message routing to the key 1111 requires only that the node's nodeID begins with 1. This property allows each overlay node to proactively maintain a small number of backup routes in its routing table. Upon detecting a failed outgoing link, a router can rapidly switch to a backup link, providing fast failover. In the background, the overlay algorithms adapt to failure by restoring (repairing) the redundancy in backup links.

To maintain high bandwidth communication, a transport infrastructure needs to not only detect failure and congestion quickly, but also quickly redirect traffic around the failures. Because each Chimera node connects to only a small number of neighbors, it can probe its overlay paths to them frequently while using a small amount of bandwidth. Since every additional hop is determined via a prefix match towards some destination (e.g., next hop node must start with prefix 111), each overlay node can maintain paths to several candidates that satisfy the routing constraint. The node redirects traffic onto backup neighbors when it detects a failure in its primary path. Meanwhile, it actively probes neighbors, requesting the location of a new neighbor to restore routing redundancy.

Chimera performs link selection using *First Reachable Link Selection* (FRLS). In this algorithm, a node η sorts

its neighbors for digit k in increasing order of latency. When a packet comes in to be routed in dimension k, η chooses its (current) closest-link neighbor. Chimera encapsulates forwarded messages in UDP within the routing overlay. End-to-end reliability and congestion control are maintained by TCP at source and destination nodes.

The application-level proxy interface to Chimera supports connecting client machines to the overlay using a modified version of the publicly available OpenVPN [14] access system. The client machine runs the unmodified OpenVPN client software to establish an encrypted tunnel to an OpenVPN server running on the Chimera ingress node. The OpenVPN client software adds entries to the client's local routing table that directs traffic for destinations served by Chimera into an established VPN tunnel.

The OpenVPN server running on the ingress router is modified to be Chimera-aware. Each server can support multiple clients. For each packet that the server receives from a client, the server looks up the overlay ID of the egress node and pushes the packet to Chimera by invoking Chimera's *push(packet, OverlayID)* function. Chimera then encapsulates the packet and tunnels it to the designated egress router. The server receives packets destined to one of its clients by calling Chimera's *pull()* function, and it then forwards the packet to the appropriate destination host over that host's VPN tunnel.

III. Hardware Routing

Unlike conventional routers, Flow State Aware (FSA) routers route and manage flows, not packets. A flow is the stream of packets from one user to another that forms a specific file transfer or conversation. A flow is uniquely identified in IPv4 by the five-tuple: source address, destination address, source port, destination port, and protocol. In IPv6 a three-tuple (flow label, source address, destination address) is used to identify the flow. The average flow is short-lived, containing only about 14 packets.

The FSA-100 hardware router supports up to 4 line cards (i.e., network interface modules) supporting either 12 auto-sensing 10/100/1000 Mbs Ethernet interface or one 10 GigE interface. OSPF and then BGP are the first supported routing protocols. Flow routers maintain performance statistics on each outgoing link. Flow statistics are exported via the industry standard *NetFlow* protocol to a computer running network management software (e.g., cflowd, flow-tools) to facilitate network-wide traffic engineering.

The FSA-100 supports rapid failover to alternate paths in the presence of a detected communication link failure between flow routers. While a conventional router typically maintains a single route for each destination, a flow router maintains a separate path for each flow. Hence, it can route each of two flows over different paths to the same destination. Having multiple active flows to the same destination is critical to obtain fast switching to an alternate path when a path fails. This contrasts with legacy routers responding to OSPF and BGP updates, which ordinarily take seconds to re-compute routes. Since a flow router can use alternate routes at any time, it maintains the best diverse near equal cost alternate route for each flow. When a path failure occurs, no re-computation is needed; the affected flow is immediately (i.e., tens of ms) switched to the backup path.

Flow routers also support network operation at high utilization. Since the rate of each flow is controlled with a flow router, the total rate being fed to a trunk is also controlled. By measuring the load on an output trunk and feeding this information back to all the input ports, the total utilization of an output port can be controlled to within 5%. We anticipate that under a normal TCP traffic mix the average utilization can be maintained above 80%, which compares favorably to the relatively lower average utilization of trunks in US carrier networks.

FSA-100 routers achieve QoS guarantees by allowing equalized load balancing among users, rapid TCP rate feedback, and guaranteed rate, loss, and delay flows for voice and video. This type of flow management is necessary in broadband networks to achieve high quality voice and video traffic. For our purposes the QoS of a flow is described by the following parameters:

- Guaranteed Rate (*GR*)
- Available Rate (*AR*)
- Burst Tolerance (*BT*)
- Delay Variance (*DV*)
- Precedence or Pre-emption Priority (*PP*)

The Network Processing Unit (NPU) of the FSA-100 router controls the QoS of all active flows by establishing QoS parameters for each flow upon setup, and adjust these parameters over the life of a flow. The data required for QoS decisions is provided to the NPU by load update messages broadcast by the Interface Modules (IM). Load update messages contain information on the average rate and number of flows for each class of traffic.

A. Establishing a Flow

The initial QoS parameters for a flow are based on the requested QoS for the flow and the QoS of the available paths. Requested QoS is determined by either an explicit request or by a rule based on a combination of fields in the first packet of a flow (i.e., a given DiffServ code may indicate Voice on IP (VoIP), which corresponds to QoS parameters of GR equaling 82 kbps and DV of less than

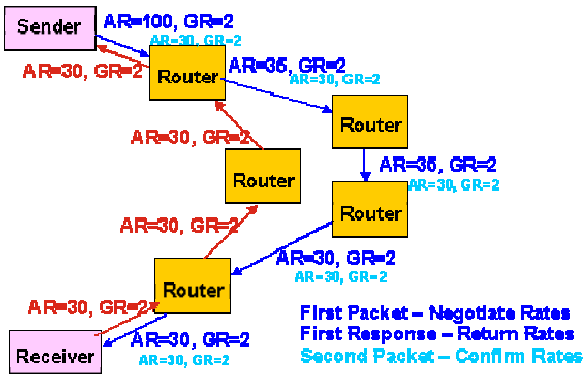

Fig. 2. QoS signaling protocol.

10 milliseconds). The available QoS of the paths that the flow could be routed over is determined by measuring the QoS parameters of all egress ports for a variety of flow classes (i.e., VoIP, video, and data). These measurements are reported to the NPU periodically. The NPU uses the requested QoS and available QoS data to select an egress port for the flow and assign granted QoS parameters to the flow.

An in-band QoS signaling protocol can be used to explicitly establish the end-to-end QoS parameters required for each new flow. This protocol has been approved in the TIA Satellite QoS group 34.1 [2] and is progressing toward approval in ITU Study Groups 11, 12 and 13. The signalling protocol is described in detail in [4]. The proposal has also been presented to IETF Working Groups for review.

Figure 2 depicts use of the signalling protocol. A sender requests a flow with available rate, guaranteed rate, delay, and precedence parameters specified. The first router reduces the available rate to what it can support. Each subsequent router in the path does the same until the message reaches the receiver. The receiver then reflects the agreed path rates to the sender, and the sender confirms them across the network to release over-commitments. A sender equipped with an Explicit Rate aware TCP stack (see IV-A) can immediately increase its sending rate to the agreed rate, in this example 32 Mbps. This rate can be maintained until the network needs to adjust the rate (up or down) due to cross-traffic.

IV. Traffic Management

CHART provides a collection of enhanced traffic management and control features designed to optimize overall system performance.

A. *'TCP-ER': Explicit Rate-Aware TCP Clients*

Besides providing end-to-end QoS via the GR, the AR computed by the flow routers can be used to significantly improve the end-to-end performance of best effort traffic flows. In particular, an explicit rate-aware TCP client can avoid the slow start phase by jumping immediately to the explicit rate (GR + AR) provided by a QoS packet within a single RTT. Besides avoiding slow-start, the AR mechanism minimizes packet loss and provides much better overall end-to-end performance for the rate-aware TCP client compared with a legacy TCP client, which relies on packet loss to infer congestion.

Legacy TCP client end-systems do not communicate transfer rate requirements to routers, but instead dynamically use TCP state information to infer the bottleneck capacity of an end-to-end path, and adjust their transmission rates to that available bandwidth. We have developed a client system capable of exploiting the availability of an end-to-end rate reservation through a network of flow routers. An initial ER-aware X86-based PC running the Fedora Core 4 operating system is used for the host. The end system uses the IPv4 version of the proposed TIA QoS protocol [2]. The TCP/IP stack has been modified to support a "fast-start" option invoked by a Linux ioctl() system call. When "fast-start" is invoked for a connection the modified TCP/IP stack will disable the TCP slow start algorithm and send at the rate specified by the QoS parameters specified in the fast-start ioctl call. The Modified TCP stack uses selective acknowledgment and retransmission (SACK) for error recovery on QoS-enabled connections.

The iptables firewall recognizes applications based on port numbers and protocol. The iptables firewall has been modified to add the ITU QoS fields to the SYN, SYN/ACK, and ACK packets for applications that have nonzero QoS parameters in the IP Tables. For applications that need QoS in only one direction, e.g., a server to client video stream, QoS can be negotiated in only one direction.

The QoS Console is a standard Linux firewall configuration tool modified to allow configuration of QoS parameters for an application. Application recognition is based on the conventional IPv4 five-tuple. If any nonzero QoS parameter is specified, the firewall will insert the required QoS signaling in any session startup (three-way handshake) and signal the protocol stack to use "fast-start."

In the absence of ER information from an ingress router, the client TCP stack operates in a conventional fashion (i.e., with slow start). It is crucial to note that legacy TCP clients will continue to operate with hardware routers. These clients will also see increases, albeit less

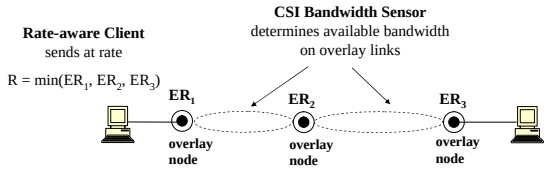

Fig. 3. An explicit rate overlay.

dramatic compared with rate-aware clients, in end-to-end bulk transfer throughput due to less variability in available bandwidth in end-to-end paths, and intelligent inter-router path management in the presence of network failures.

B. An Explicit Rate Overlay

The AR mechanism (with GR $= 0$) extends to paths that do not consist exclusively of flow routers in two scenarios:

1) The estimated *available bandwidth* along a subpath of legacy routers between two flow routers on a given path would be used by the upstream flow router to compute the AR value corresponding to the "virtual" link between the two flow routers.
2) A software overlay router along a given path could compute the AR value for an outgoing overlay link similarly to a flow router and write the AR information onto QoS packets traversing the link.

In both cases, the available bandwidth information for a given Internet path would be provided by the CHART sensing system.

The second case corresponds to what we call an *explicit rate (ER) overlay*. Figure 3 depicts how the ER overlay allows the software overlay routing system to emulate the available rate feature provided by the flow routers. IP QoS flows with GR $= 0$ can be supported by an ER overlay even in the absence of the hardware-based flow routers. Within the ER overlay, a given overlay node computes an AR value corresponding to each outgoing overlay link using available bandwidth estimates obtained from the sensing infrastructure and internal packet queue lengths. The AR value for an overlay link is reported to the flows traversing the link in accordance with the TIA IP QoS protocol [2].

The algorithm employed by the overlay node to compute the AR value need not be the same as the algorithm used in the hardware-based flow routers. We are currently experimenting with an ER control algorithm based on the NEC rate control algorithm for Available Bit Rate (ABR) services in ATM networks [16]. This algorithm does not require maintenance of per-flow state, has good performance characteristics, and is stable. Besides supporting the IP QoS flows (with zero GR value), the ER overlay can also support Quick-Start TCP, an enhancement to TCP proposed within the IETF [15]. Quick-Start TCP allows a flow to jump to an large initial congestion window size via a signaling mechanism similar to the TIA IP QoS protocol, but does not provide a full-fledged congestion control mechanism.

V. System Performance

The CHART system has been designed increase end-to-end bulk file transfer throughput performance by a factor ten in a challenging operational WAN environment. The system is currently being tested on 3 distinct testing platforms – *Netbed* [12], *DETER* [13], and *PlanetLab* [8]. Netbed (i.e., Emulab) is primarily used to emulate networks with link impairments. Certain security tests – such as robustness to distributed denial-of-service attack – are being performed in the contained environment of the DETER testbed. PlanetLab testing is being used to validate system operation at scale, test network sensors in the presence of unknown cross-traffic, and validate system operation in the presence of unanticipated network behaviors.

To verify the performance of control plane enhancements, DARPA has created a set of three network test scenarios [10, 11]. Each scenario will introduce up to 10 separate network impairments or defects, such as an excessively high packet loss on a link, or temporary link failure. Bulk data transfer tests will be performed using both *ftp* and *http*. Figure 4 shows one proposed scenario, where a naval fleet with inter-ship wireless radio frequency communications has a choice of two available satellite communication links to access a web server in the continental United States (CONUS). The two satcom options are markedly different, perhaps representing low and high earth orbit satellites. A third connectivity option is to forward packets to another ship as an intermediary when beneficial to exploit that intermediate ship's higher quality satellite uplink. Users on each ship may be accessing the server simultaneously, and time-varying link impairments occur on both RF and satellite connections.

We have used the *ns-2* network simulator to determine the baseline performance of each test scenario prior to the deployment of CHART control plane enhancements. Further, our enhancements are being tested in an Emulab prototype of each test topology. Figure 5 depicts the Emulab network topology we have constructed to model the Naval fleet test configuration.

Using Emulab we have measured the performance of CHART software components both individually and when combined to form a complete system. In one set of experiments we set out to examine CHART's end-to-end TCP

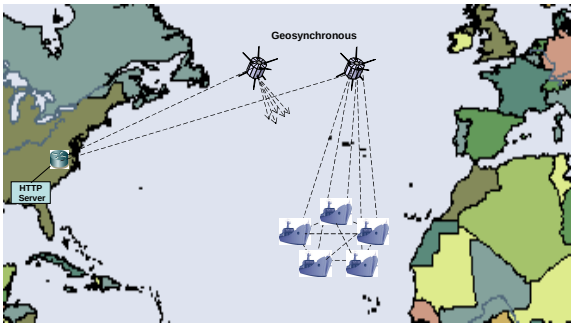

Fig. 4. Naval fleet test scenario.

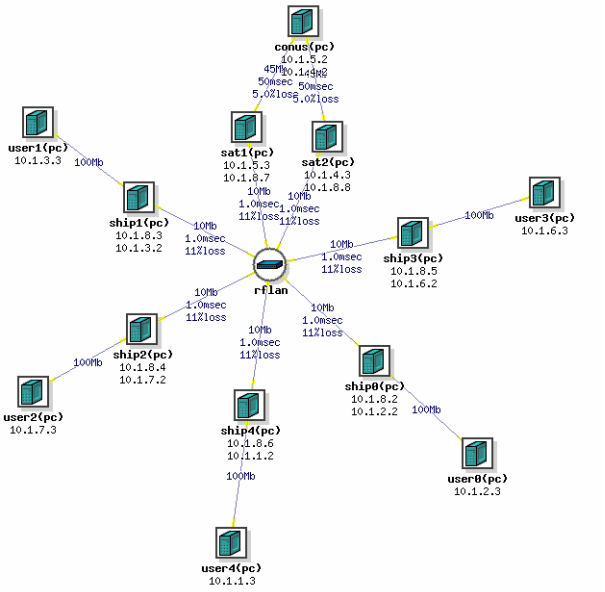

Fig. 5. Netbed emulation of Naval fleet test configuration.

throughput when facing severely impaired communication links between Chimera overlay nodes. In this experiment 2 disjoint paths were available between ingress and egress overlay nodes, with a single source node attached to the overlay ingress, and a single destination node attached to the overlay egress. All communication links in the network operated at 100 Mbs. Each of the two available paths contained a wide-area network link with identical packet loss and propagation delay, with unidirectional packet loss ranging from 0-5% and unidirectional delay from 0-100 ms.

We deployed a 6 node Chimera overlay network, equipping the source and destination nodes with OpenVPN to connect to the overlay. Figure 6 shows a comparison of the average end-to-end TCP throughput through the overlay measured by *iperf* with a 64 Kbyte TCP window size for both Legacy TCP and for TCP-ER. No additional cross-traffic was applied, and the Available Rate

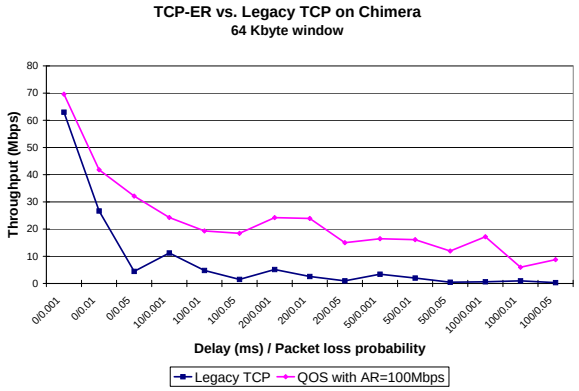

Fig. 6. Throughput for TCP-ER vs. legacy TCP with a 64 Kbyte TCP window across an overlay with varying packet loss and delay.

(AR) set by the TCP-ER protocol stack was 100 Mbs. The figure clearly shows that TCP-ER dramatically outperforms Legacy TCP as propagation delay and packet loss increases. Figure 7 shows a comparison of the average end-to-end TCP throughput measured when we increased the TCP window size to 1 Mbyte.

Though not shown in theses figures, the Chimera overlay provided the end-to-end system with robustness to link failure, typically re-routing packets to the alternate path in approximately one-third the time required for OSPF to respond to the link failure. In the presence of link failures this rapid re-routing can and does significantly increase throughput performance. Figures 6 and 7 also reveal the performance 'cost' of overlay routing. In the absence of loss and delay, the expected TCP throughput of this network would be roughly 94 Mbs. However, due to the latencies incurred in having TCP segments forwarded by multiple Chimera nodes, and encapsulated 3 times – twice by OpenVPN tunnels and once by Chimera – the maximum performance realized is approximately 72 Mbs. Of course, this bottleneck is typically not encountered in target networks where losses and delays are non-negligble and TCP is limited to sending at considerably lower rates.

VI. Summary

The combination of two complementary routing solutions with a common, pervasive, network-wide sensing infrastructure on top of a network-wide virtual machine creation service is the defining contribution of CHART. This approach permits the gradual introduction of new routing technology on the network, where overlay nodes on the existing infrastructure provide the immediate benefits of adaptive routing. Both hardware and software routers are

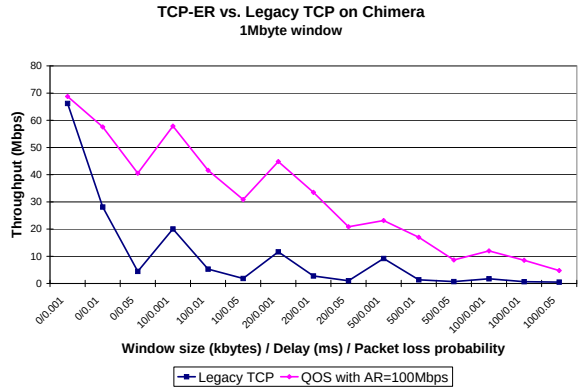

Fig. 7. Throughput for TCP-ER vs. legacy TCP with a 1 Mbyte TCP window across an overlay with varying packet loss and delay.

guided by a strong awareness of both local and remote network conditions obtained through network sensors and a sensor data propagation service. Inter-router communications and sensor information are certified by an innovative security architecture. The cost of the solution is low (approximately the cost of an extra PC per subnet) for all but very high-bandwidth (gigabit and beyond) links.

The CHART system offers a new approach to the problem of improving end-to-end throughput performance between CONUS and forward-deployed units, combining software-assisted alternate path routing, decentralized performance measurement and monitoring, and the strategic deployment of a new generation of advanced routers. The combination of network sensing and adaptive routing has been shown in preliminary tests to increase throughput in the presence of bad links by an order of magnitude or more for both pure hardware and pure software implementations. A unique and compelling aspect of CHART is that it permits the gradual introduction of new routing technology on the network, where overlay nodes on the existing infrastructure provide the immediate benefits of adaptive routing. Hardware routers may be judiciously added to a network to accommodate transmission rates up to 10 Gbs. Overlay nodes also facilitate the introduction of new network applications beyond the initial use of intelligent routing.

CHART's approach not only solves the end-to-end performance problem sought by the DARPA Internet Control Plane program, but lays a foundation for future network-wide applications. The reason for this is that once a computational overlay is deployed – in this case to support software routing – the overlay can be exploited for a variety of other new, decentralized applications.

REFERENCES

[1] J. Crowcroft, I. Phillips, *TCP/IP and Linux Protocol Implementation*, Wiley, New York, 2002.

[2] L. Roberts, *TIA TR-34.1.02/12.04.05: QoS Signalling for IPv6 QoS Support*, TIA TR-34.1.7 Working Group (IP on Satellite), 2005.

[3] Lawrence G. Roberts, "The Next Generation of IP - Flow Routing", *Proceedings of SSGRR 2003*, Italy, 2003.

[4] Lawrence G. Roberts, "Major Improvements in TCP Performance over Satellite and Radio," *Proceedings of Milcom 2006*, Washington DC, 2006.

[5] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea, Anthony D. Joseph, John Kubiatowicz, "Tapestry: A Global-scale Overlay for Rapid Service Deployment", *IEEE JSAC: Special Issue on Recent Advances In Service Overlay Networks*, vol. 22, num. 1, pp. 41-53, January, 2004.

[6] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Anthony D. Joseph, and John D. Kubiatowicz, "Exploiting Routing Redundancy via Structured Peer-to-Peer Overlays", *Proceedings of the 11th IEEE ICNP*, Atlanta, Georgia, October, 2003.

[7] T. Roscoe, L. Peterson and S. Karlin, "A Simple Common Sensor Interface for PlanetLab",*PlanetLab Design Note PDN–03–010*, PlanetLab Consortium, May 2003.

[8] PlanetLab, `http://www.planet-lab.org`.

[9] Larry Peterson, Tom Anderson, David Culler, and Timothy Roscoe, "A Blueprint for Introducing Disruptive Technology into the Internet", *1st ACM HotNets Workshop*, October 2002.

[10] J. Meagher, *SAIC Design Document for Performance Lab Testbed*, 2005.

[11] J. Meagher, *SAIC Test Report for Performance Lab Testbed*, 5 August 2005.

[12] Emulab Network Emulation Testbed, `http://www.emulab.net`.

[13] The DETER Testbed: Overview, `http://www.isi.edu/deter/docs/testbed.overview.htm`.

[14] OpenVPN, `http://openvpn.net`.

[15] A. Jain, S. Floyd, M. Allman, and P. Sarolahti, "Quick-Start for TCP and IP," Internet-draft draft-ietf-tsvwg-quickstart-00.txt, work in progress, May 2005.

[16] A. Kolarov and G. Ramamurthy, "A Control-Theoretic Approach to the Design of an Explicit Rate Controller for ABR Service," *IEEE/ACM Trans. on Networking*, vol. 7, no. 5, Oct. 1999.

[17] B.L. Mark and G. Ramamurthy, "Real-time Traffic Characterization for Quality-of-Service Control in ATM Networks," *IEICE Transaction on Communications*, Vol. E81-B, No. 5, pp. 832-839, July 1998.

[18] Gianluca Iannaccone, Chen-Nee Chuah, Richard Mortier, Supratik Bhattacharyya, Christophe Diot, "Analysis of link failures in an IP backbone", *Proceedings of the Internet Measurement Workshop*, Marseille, France, Nov 2002.

[19] P. Yalagandula, P. Sharma, S. Banerjee, S. Basu, SJ Lee, "S^3: A Scalable Sensing Service for Monitoring Large Networked Systems," *Proceedings of the Internet Network Management Workshop*, Pisa, Italy, Sept. 2006.

[20] David Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris, "Resilient Overlay Networks", *Proceedings of SOSP 2001*, October 2001.