Sigcomm 2002 Student Posters

Measurement

1. Baek-Young Choi, Jaesung Park, and Zhi-Li Zhang, *University of Minnesota*, **Adaptive Packet Sampling for Flow Volume Measurement**

2. Cristian Estan, George Varghese, Mike Fisk, *University of California San Diego*, **Counting the number of active flows on a high speed link**

3. Krishna P. Gummadi, Stefan Saroiu and Steven D. Gribble, *University of Washington,* **King: Estimating Latency between Arbitrary Internet End Hosts**

Network Architecture and Topology

4. Pablo Molinero-Fernandez, and Nick McKeown, *Stanford University*, **The Performance of Circuit Switching in the Internet**

5. Danica Vukadinovic, Thomas Erlebach, Polly Huang, Maurice Ruegg Roman Schilter, *Swiss Federal Institute for Technology Zurich (ETHZ)*, **Real and Generated Internet AS Topologies: Structure, Spectrum, Robustness**

Overlay Networks, Peer to Peer Systems, and Application Support

6. Sachin Agarwal, Avi Yaar, David Starobinski, Ari Trachtenberg, *Boston University*, **Fast Network Synchronization**

7. Dejan Kostic, Adolfo Rodriguez and Amin Vahdat, *Duke University*, **ACDC: Scalable and Adaptive Two-metric Overlays**

8. Joanna Kulik, *Massachusetts Institute of Technology*, **Encoded Meta-Data Formats for Internet Subscription Systems**

9. Angelos Stavrou. Dan Rubenstein, Sambit Sahu, *Columbia University*, **A Lightweight, Robust P2P System to Handle Flash Crowds**

Provisioning

10. Frances Chang, Wu-chang Feng, Wu-chi Feng, Jonathan Walpole, *Oregon Health & Science University*, **Provisioning On-line Games: A Traffic Analysis of a Busy Counter-Strike Server**

11. Peerapon Siripongwutikorn, Sujata Banerjee, David Tipper, *University of Pittsburgh*, **Adaptive Bandwidth Control for Efficient Aggregate QoS Provisioning**

Queue Management

12. Debojyoti Dutta, Ashish Goel, John Heidemann, *University of Southern California*, **Oblivious AQM and Nash Equilibria**

13. Wesley M. Eddy, Mark Allman, *Ohio University*, **A Comparison of RED's Byte and Packet Modes**

14. Ganesh Gopalakrishnan, Sneha K. Kasera, *University of Illinois, Urbana Champaign*, **Robust Rate Based Congestion Control**

Routing and routers

15. Armando L. Caro Jr., Janardhan R. Iyengar, Paul D. Amer, Gerard J. Heinz, Randall R. Stewart, *University of Delaware*, **Using SCTP Multihoming for Fault Tolerance and Load Balancing**

16. Ji Li, Hauyang Liu, Karen Sollins, *Massachusetts Institute of Technology*, **AFBV: A Scalable Packet Classification Algorithm**

17. Ken Yocum, Darrell Anderson, Jeff Chase, Amin Vahdat, *Duke University*, **Anypoint: Extensible Transport Switching on the Edge**

Tools

18. Martin Casado, Vikram Vijayaraghavan, Guido Appenzeller and Nick McKeown, *Stanford University*, **The Stanford Virtual Router - A teaching tool and network simulator**

19. Brian White, Shashi Guruprasad, Mac Newbold, Jay Lepreau, Leigh Stoller, Robert Ricci, Chad Barb, Mike Hibler, Abhijeet Joglekar, *University of Utah*, **Netbed: An Integrated Experimental Environment**

20. Ken Yocum, Kevin Walsh, Amin Vahdat, Priya Mahadevan, Dejan Kostic, Jeff Chase, David Becker, *Duke University*, **Scalability and Accuracy in a Large-Scale Network Emulator**

Wireless and Ad Hoc Networks

21. L. Balzano, V. Gambiroza, Y. Liu, P. Yuan, E. Knightly, S. Shaefor, *Rice University*, **DVSR: a High-performance Metro Ring Protocol**

22. Karthik Dantu, Shyam Kapadia,, Rishi Sinha, Ahmed Helmy, *University of Southern California*, **Modeling of Mobility-Induced Losses in MANETs (MILMAN)**

23. Glenn Judd and Peter Steenkiste, *Carnegie Mellon University*, **Fixing 802.11 Access Point Selection**

24. Nitin Nahata, Priyatham Pamu, Saurabh Garg, Ahmed Helmy, *University of Southern California*, **Efficient Resource Discovery for Large Scale Ad hoc Networks using Contacts**

Adaptive Packet Sampling for Flow Volume Measurement

Beak-Young Choi, Jaesung Park, Zhi-Li Zhang

http://www.cs.umn.edu/networking/sampling/samplingProject.htm
Department of Computer Science and Engineering, University of Minnesota
E-mail: {choiby, jpark, zhzhang}@cs.umn.edu

In this study, we have addressed the problem of flow volume measurement using packet sampling approach. Traffic measurement and monitoring serve as the basis for a wide range of IP network operations, management and engineering tasks. Particularly, *flow-level* measurements are required for applications such as traffic profiling, usage-based accounting, traffic engineering, traffic matrix and QoS monitoring. With today's high-speed (e.g., Gigabit or Terabit) links, however, capturing every packet is no longer feasible due to the excessive overheads it incurs on linecards or routers. As a scalable alternative, both the Internet IETF working groups IPFIX (IP Flow Information Export) and PSAMP (Packet Sampling) have recommended the use of packet sampling.

The foremost and fundamental question regarding sampling is its accuracy. This is especially pertinent in the Internet, where traffic is known to fluctuate dynamically and unpredictably. Inaccurate packet sampling not only defeats the purpose of traffic measurement and monitoring, but worse, can lead to wrong decisions by network operators. Excessive oversampling should also be avoided for the measurement solution to be scalable. Therefore, it is important to *control* the accuracy of estimation so as to balance the *trade-off between the utility and overhead of measurements*. Given the dynamic nature of network traffic, *static* sampling does not always ensure the accuracy of estimation, and tends to oversample at peak periods when economy and timeliness are most critical.

Packet sampling for *flow-level measurements* is a particularly challenging problem. One issue is the diversity of flows: flows can vary drastically in their volume. The dynamics of flows is another issue: flows arrive at random time and stay active for a random duration; the rate of a flow (i.e., the number of packets generated by a flow per unit of time) may also vary over time, further complicating the matter of packet sampling.

How can we ensure accuracy of measurement of *dynamic* flows? How many packets does one need to sample in order to produce flow measurements with a *pre-specified error bound*? How to decide on a sampling rate to avoid excessive oversampling while ensuring accuracy? How to perform sampling procedure and estimate flow volume? How easily can it be implemented at line speed? To answer these questions, in this paper we advance a theoretical framework and develop an *adaptive* packet sampling technique using *stratified random sampling*.

The technique is targeted for *accurate* (i.e., with bounded sampling errors both in packet and byte count) estimation of *large* or elephant flows, since a small percentage of flows are observed to account for a large percentage of the total traffic. We classify flows based on *proportion of packet count* to the total count over a time interval that encompasses the flow duration. A flow is referred to as an elephant flow in our study if its packet count proportion is larger than a pre-specified threshold. Since the proportion is defined over the time interval enclosing a flow, it eliminates the rate fluctuation impact. This definition of elephant flows captures flow characteristics of packet count, byte count as well as burstiness.

We tackle the problem of flow dynamics using *stratified random sampling*. The proposed technique uses *predetermined non-overlapping time blocks* called strata. For each stratum, it samples packets with the same rate. The sampling rate is set to bound the estimation error of the smallest elephant flow. At the end of each stratum, flow statistics are estimated and updated. The predetermined time blocks enable us to estimate flow volume without knowing flow arrival times and their durations. From each flow's point of view, it is stratified in fixed time. At the end of the last time block enclosing the flow, its statistics are summarized into a single estimation record.

A time block is the minimum time scale over which an elephant flow (packet count proportion) is determined. It is also minimum time scale over which the sampling rate can be adjusted. In order to achieve a desired accuracy, a certain *minimum* number of packets must be sampled during the sampling frame that encompasses an elephant flow duration. Given the arbitrary length of elephant flow duration, the sampling frame could be one block or a few consecutive blocks in the stratified sampling. To ensure that we collect enough samples of each elephant flow, we sample the minimum required number of packets every block. The minimum required number of samples is computed at the beginning of each block to accommodate dynamic changes in the total traffic.

Using real network traffic traces, we demonstrate that the proposed technique indeed produces the desired accuracy of flow volume estimation, while at the same time achieving significant reduction in the amount of packet samples as well as flow cache size.

Counting the number of active flows on a high speed link

Cristian Estan, George Varghese, Mike Fisk
Computer Science and Engineering Department
University of California San Diego
cestan,varghese,mfisk@cs.ucsd.edu
http://www.cs.ucsd.edu/users/cestan

Abstract

Internet links operate at high speeds, and past trends predict that these speeds will continue to increase rapidly. Routers and Intrusion Detection Systems that operate at up to OC-768 speeds (40 Gigabits/second) are currently being developed. In this paper we address a basic function common to several security and measurement applications running at line speeds: *counting the number of distinct header patterns (flows) seen on a high speed link in a specified time period.* For example one can detect port scans by counting the number of connections opened by the suspected port scanner and one can ascertain that a denial of service attack is in progress by counting the number of distinct (fake) source addresses that send packets to the suspected victim. We provide algorithms solving the flow counting problem using extremely small amounts of memory. This can translate into savings of scarce fast memory (SRAM) for hardware implementations. It can also help systems that use cheaper DRAM to allow them to scale to larger instances of the problem. The reduction in memory is particularly important for network security applications such as detecting port scans and DoS attacks that need to run a separate instance of the algorithm for each suspected attacker or victim. Our algorithms can be implemented in hardware at wire speeds (8 nsec per packet for OC-768) using simple CRC based hash functions, multiplexers and SRAM. They access at most one or two memory locations per packet. We expose the behavior of simple building blocks and provide a family of *customizable* counting algorithms that can be adapted to the requirements of various applications. This allows our algorithms to use less memory than the best known counting algorithm, probabilistic counting, to provide the same accuracy.

- Virtual bitmap is well suited for triggers (which need only be accurate around a threshold value) such as detecting DoS attacks, and uses 292 bytes to achieve an error of 2.619% compared to 5,200 bytes for probabilistic counting.
- Our "no assumptions" counting algorithm, *multiresolution bitmap* algorithms uses only 2143 bytes to count up to 100 million flows with an average error of 3%. Its accuracy is slightly better when the number of flows is small, while the accuracy of probabilistic counting using the same memory is much worse.
- Our *adaptive bitmap* that exploits stationarity in the number of flows can count the number of distinct flows on a link that contains anywhere from 0 to 100 million flows with an average error of less than 1 % using only 2 Kbytes of memory. Probabilistic counting needs eight times more memory to achieve the same accuracy.
- We used our *triggered bitmap* to replace the port scan detection component of the popular intrusion detection system Snort. Doing so reduced the memory usage from 89 Mbytes to 6.4 Mbytes (with an average error of 13.5%) as measured on a 10 minute trace. Probabilistic counting uses 23 Mbytes.

This work was made possible by a grant from NIST for the Sensilla Project.

King: Estimating Latency between Arbitrary Internet End Hosts *

Krishna P. Gummadi, Stefan Saroiu, Steven D. Gribble
Department of Computer Science & Engineering
University of Washington Seattle, WA, USA, 98195-2350
{tzoompy,gummadi,gribble}@cs.washington.edu

The ability to estimate network latencies accurately between arbitrary Internet end hosts would enable wide-area measurement studies and applications, such as investigating routing path inefficiencies on a wide-scale [1] or constructing topologically sensitive overlay networks. In this poster we present King [2], a tool that accurately, quickly and cheaply estimates the latency between arbitrary end hosts by using existing DNS infrastructure in a novel way. Our approach is inspired by tools like Sting [3] and T-BIT [4], which show that it is possible to use existing protocols in unanticipated ways to obtain results that were previously intractable.

Our technique relies on two observations. First, given a pair of end hosts to be measured, in most cases it is possible for King to find their authoritative DNS name servers that are topologically close to the end hosts. Second, given a pair of DNS name servers, King can accurately estimate the latency between them using recursive DNS queries. Thus, King is able to use the measured latency between the name servers as an estimate of the latency between the end hosts. Although this estimate will inevitably suffer from inaccuracies, our extensive evaluation demonstrates that this estimation error is small (less than 20% error in over three-quarters of generated estimates).

Compared to previous approaches like IDMaps [5] and GNP [6], King has two primary advantages (1) deployability and (2) accuracy.

1. *Deployability:* Unlike IDMaps, King does not require the deployment of additional infrastructure, and unlike GNP, King does not require end hosts to agree upon a single set of landmarks or share the coordinates with other end hosts. Because King uses existing DNS infrastructure, it scales naturally in terms of the number hosts that could be measured. Our evaluation indicates that King can be used to estimate latencies between arbitrary IP hosts more than 75% of time in the Internet today.

2. Accuracy: Unlike IDMaps and GNP the accuracy of King is not affected by the placement of measurement points (tracers or landmarks) in the Internet. King's estimates are based on direct online measurements rather than extrapolation from latencies of a subset of paths measured offline. Our evaluation shows that the error in King latency estimates between end hosts is significantly smaller than that of IDMaps, while the error in its estimates between name servers is negligible. Further, we show that King can identify those of its own estimates that are likely to be inaccurate. These characteristics make King an ideal tool for a number of wide-area measurement studies that involve measuring latencies accurately between a large number of geographically distant end hosts.

King is also fast and lightweight requiring the generation of only a few packets to produce an estimate. Our evaluation also demonstrates that (a) King preserves order among its latency estimates; a feature useful in identifying a close server from a group of server replicas and (b) authoritative name servers are located topologically close to their end hosts in a majority of scenarios (i.e., latencies as measured from a far away host to both the end host and its name server are very similar); this explains the high accuracy of King. Finally, we test the feasibility of using King for measurement studies by performing Detour study [1] on a larger scale. The results we obtained are consistent with the earlier Detour study. We were able to obtain our results with far greater ease as, unlike the previous study, we were not restricted to the publicly available traceroute servers and their usage policies. We also believe that the simplicity of our King tool will enable researchers to use it in myriads of unanticipated ways like the more popular Ping tool.

*This work was supported by National Science Foundation under ITR grant No. CCR-0121341. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] "The Detour web page." www.cs.washington.edu/research/networking/detour/.

[2] "The King web page." www.cs.washington.edu/homes/gummadi/king.

[3] "The Sting web page." www.cs.washington.edu/homes/savage/sting/.

[4] "The TBIT web page." www.icir.org/tbit/daxlist.txt.

[5] "The IDMaps web page." idmaps.eecs.umich.edu/.

[6] "The GNP web page." www-2.cs.cmu.edu/~eugeneng/research/gnp/.

The Performance of Circuit Switching in the Internet

Pablo Molinero-Fernández, and Nick McKeown, Stanford University

{molinero,nickm}@klamath.stanford.edu

ABSTRACT:

Recently there has been renewed interest in circuit switching because of the ease of building very high capacity optical circuit switches. This poster studies the performance, as seen by the end user, of an Internet that uses circuit switching instead of, or in addition to, packet switching. TCP represents over 90% of current Internet traffic, which makes our usage of the network very connection oriented. Therefore, the key performance metric for the end user is the completion time for each connection, which we refer to as response time.

On the face of it, simply considering circuit switching as an alternative would seem a pointless exercise; the Internet is packet switched, and was deliberately built that way to enable the efficiencies afforded by statistical multiplexing, and thus it should provide a response time that is faster than that of circuit switching. However, link utilization is low, and falling, particularly at the core of the Internet, which means that statistical multiplexing is less important than it once was.

In this poster, we explore the performance of a network where a new circuit is created for each application flow, with particular emphasis on the response time experienced by users. We use simple M/G/1 and M/G/N queues to model application flows in both packet switched and circuit switched networks, as well as ns-2 simulations to validate the results. We conclude that because of high-bandwidth long-lived flows, it does not make sense to use circuit switching in shared access or local area networks. At the same time, our results suggest that in the core of the network, where access links limit the maximum flow rate, and where high capacity is needed most, there is little or no difference in performance between circuit switching and packet switching. Given that circuit switches can be built to have higher capacity than packet switches, this suggests that a circuit switched core warrants further investigation.

There are other performance metrics that are also very important when deciding whether circuit switching is a good technology for the Internet, such as scalability, simplicity, cost, reliability and availability, Quality of Service (QoS), or traffic engineering. We have used such criteria in other studies, but they will not be discussed in the poster with great detail given the space constraints. The end result is that we greatly benefit from the use of circuit switching in the backbone of the Internet.

Our current use of the network is very connection oriented, and, thus, it fits well with a network core that is circuit switched, while the edges use packet switching. We may just use a gateway that maps packet switched flows (e.g. TCP connections) to circuits. This gateway creates and destroys circuits as flows come and go. Then the deployment of circuit switching in the core does not require any change in existing end hosts and routers, which will continue to use IP the same way as today, as the gateways will hide the existence of circuits in the core. With this network architecture, circuit switched clouds can be inserted gradually, without requiring a flag day in the Internet.

FOR MORE INFORMATION:

`http://klamath.stanford.edu/TCPSwitching/`

Real and Generated Internet AS Topologies: Structure, Spectrum, Robustness *

Danica Vukadinović, Thomas Erlebach, Polly Huang, Maurice Rüegg, Roman Schilter
Computer Engineering and Networks Laboratory (TIK)
Swiss Federal Institute of Technology (ETH), Zurich, Switzerland
E-mail:{vukadin,erlebach,huang}@tik.ee.ethz.ch; {mrueegg,rschilte}@ee.ethz.ch

The Internet AS (Autonomous System) topology structure is a subject of high interest as performance of Internet protocols and algorithms can depend heavily on the topology. Besides other reasons, a good model for the Internet topology may lead to a realistic generator, used for simulations. Power-law degree Internet topology generators, now state-of-the-art, result in different kinds of graphs. It has already been observed that power-law distribution creates hierarchy, although the type of hierarchy differs from model to model. Also, one should take into account the influence of BGP policies by considering them as part of the AS topology model. We looked at the real and generated Internet Topology in order to find relevant and computationally simple metrics to evaluate differences. Preliminary results using the Normalized Laplacian Spectrum, which can be calculated in polynomial time, show that it is a good candidate to differentiate classes of topologies.

Some of the questions we are interested in are:

– **What can we conclude about topology from the normalized spectrum of the graph?**

Preliminary results: Observation of the difference between the multiplicities of the eigenvalue 1 in the Normalized Laplacian Spectrum of graphs created by Inet 2 (Inet Topology Generator, University of Michigan) and of corresponding AS graphs lead us to a new hierarchical representation of the AS graph. We partition the graph into 4 components: P, Q, R and I. Taking into account BGP policies, we confirmed our conclusions about the functionality of each component (roughly, the edge and the core). Following the components' evolution over time, we could extract some trends: more nodes are added to the edge, but more new links are added to the core. Experiments on other multiplicities showing occurrences of specific subgraphs, convexity and concavity of the parts of the spectrum are still part of our research.

– **What is the relevance of other metrics from graph theory?**

Preliminary results: Vertex Cover, Maximum Independent Set and Maximum Clique are some examples of *NP*-complete problems, that are efficiently solvable on the Internet AS topologies. Although primarily of theoretical interest, some of these metrics can be attractive for practical reasons too, being efficiently computable. A positive example is the optimal placement of the DDoS protection filter in the Vertex Cover set.

– **How can we measure robustness in the whole network and in its parts? Can we improve "topological" robustness of a part of the network without overdesign?**

Preliminary results: We looked at the number of node-disjoint paths from Oregon to all ASs in the graph model and in the data obtained from BGP tables with respect to the direction of traffic forwarding. We have to stress that the information from the BGP tables is inaccurate and (because of *NP*-completeness of the problem) heuristics are used for the measurements. However, there is a significant difference between the two results.

Remark: Some of the preliminary results on the spectrum of the AS graphs are already published (more information could be found on http://www.tik.ee.ethz.ch/∼the/topology/).

*Partially supported by European Commission - Fet Open project COSIN IST-2001-33555, with funding provided by BBW Switzerland.

Fast Network Synchronization

Sachin K Agarwal, Avi Yaar, David Starobinski, Ari Trachtenberg
{ska, ayaar, staro, trachten}@bu.edu
Department of Electrical and Computer Engineering, Boston University

Much of the popularity of mobile computing and handheld devices can be attributed to their ability to deliver information to users on a seamless basis. Periodic synchronization of data between various devices is an enabling technology for this computing paradigm and an essential feature of any effective mobile and heterogeneous network architecture. We propose a peer-to-peer synchronization algorithm based on recent information-theoretic advances [3] and demonstrate an implementation of that algorithm through an improved PC-PDA synchronizer.

PalmPilot PDAs (Personal Digital Assistants) synchronize their data with that on a host PC either locally over a USB or serial connection, or remotely through a TCP/IP network - using a protocol known as HotSync [1, 2]. This protocol has two modes of operation: *FastSync*, in which stored modification flags identify the exact records that need to be exchanged; and *SlowSync*, in which all of the PDA s records are transferred to the PC for an item-by-item comparison to assess their modification status. FastSync represents the ideal case for network synchronization, where the differences between the synchronizing computers are known (through the modification flags) *a priori*. FastSync is limited to a two-host system, however, because each host maintains only one set of flags, so only differences between two specific hosts can be detected. The more generic SlowSync is representative of present-day network synchronization technology, which sacrifices network bandwidth and efficiency for robustness and simplicity.

The major problem of the SlowSync algorithm is that it does not scale with the size of the synchronizing data sets. This vulnerability will become more and more costly as data storage capacity outpaces the increase in network bandwidth. To address this scalability issue, we have developed and analyzed an algorithm termed CPISync (Characteristic Polynomial Interpolation Synchronization), based on a recent solution to the set reconciliation problem given in [3]. The CPISync algorithm uses interpolation of a rational function whose components can be factored to identify the differences between data sets stored on any two synchronizing computers. Since the number of sample points needed to correctly perform the interpolation depends only on the number of *differences* between the two synchronizing data sets, the communication complexity of CPISync is proportional to the differences between the sets. The algorithm's main drawback, however, is that the interpolation and factorization steps demand a cubic computation time (in the number of differences). The impact of this is lessened by the fact that the computation can be distributed asymmetrically according to the relative computing power of the synchronizing machines [1, 4].

We have implemented and deployed the CPISync algorithm in a PC-PalmPilot synchronization system, which we feel accurately models a peer-to-peer link in a heterogeneous network. Our PDA application, a memo entry program called *SyncMemo*, uses CPISync to synchronize data with its PC counterpart *PCMemo*. In tests against the PalmPilot s built-in *MemoPad* application, which uses the SlowSync algorithm, *SyncMemo* consistently synchronized faster, except in cases where there was a large number of differences between the synchronizing data sets. We generated results that helped to determine the communication and computation tradeoffs between CPISync and SlowSync in a variety of potential applications. The results from our prototype can be applied to reducing communication bottlenecks for many network protocols in large, heterogeneous systems, including the Domain Name System (DNS), rsync, heterogeneous mobile device synchronization, resource discovery and link state routing.

Website: *http://ipsit.bu.edu/nislab*

This work was supported in part by NSF Career Grant No. CCR-0133521

REFERENCES

[1] S. Agarwal, *Data Synchronization in Mobile and Distributed Networks,* Masters Thesis, July 2002.

[2] S. Agarwal, D. Starobinski and A. Trachtenberg, *On the Scalability of Data Synchronization Protocols for PDAs and Mobile Devices*, IEEE Network (Special Issue on Scalability in Communication Networks), Vol. 16, No.4, pp.22-28, July/August 2002.

[3] Y. Minsky, A. Trachtenberg and R. Zippel, *Set Reconciliation with Nearly Optimal Communication Complexity*, International Symposium on Information Theory, Washington D.C., June 2001.

[4] A. Trachtenberg, D. Starobinski, and S. Agarwal, *Fast PDA Synchronization using Characteristic Polynomial Interpolation*, Proceedings of the IEEE INFOCOM 2002, pp. 1510-1519, New York.

ACDC: Scalable and Adaptive Two-metric Overlays

Dejan Kostić, Adolfo F. Rodriguez, and Amin Vahdat *
Department of Computer Science, Duke University
{dkostic,razor,vahdat}@cs.duke.edu
http://issg.cs.duke.edu

1 Motivation

Currently, application-layer multicast is used for a broad range of applications, including multimedia distribution, event notification, and update propagation among wide-area replicas. However, the proper overlay topology remains an open problem, as higher performance typically incurs higher cost. The goal of our work is to dynamically build low-cost scalable overlays that meet performance targets. Our challenges include scalability, adaptivity, approximate information, and two independent, opposing metrics (for example, cost and delay).

2 ACDC Overlays

ACDC (Adaptive low-Cost, Delay Constrained) algorithm builds and maintains overlay distribution trees with the following characteristics: i) scalable, ii) degree-constrained, iii) delay-constrained, iv) low-cost, v) adaptive, and vi) self-organizing. The problem of finding a delay-constrained minimum cost spanning tree (DCMST) is NP-complete even when using global knowledge. Therefore, we approximate the global solution in a decentralized manner using partial information.

ACDC exports a knob that allows applications to trade cost for performance based on application and network characteristics. The knob traverses the space between the minimum cost spanning tree (MST) built over the cost metric and the shortest path tree (SPT) that would be built over the delay metric.

*This research is supported in part by the National Science Foundation (EIA-9972879, ITR-0082912), Hewlett Packard, IBM, Intel, and Microsoft. Vahdat is also supported by an NSF CAREER award (CCR-9984328).

3 Achieving Scalability

ACDC achieves scalability through the probabilistic distribution of probe sets to each node once per epoch that achieves a total ordering among all participating overlay nodes. An epoch consists of two phases: one distribute phase to transmit data from the root to all overlay participants (data is distributed down the tree) and a second collect phase where each participant successively updates its parent's status information (data is aggregated up the tree). During the distribute phase, each node sends to its children a probe set of remote nodes (O(lg n) in size by default). The contents of the probe set are constructed from candidate sets gathered during the previous collect phase. Each node A performs probes to members of its probe set to determine if a remote node B exists that would deliver better cost, delay or both to A and its descendants.

A total ordering among nodes distributed in the probe set ensures that two nodes cannot perform simultaneous overlay transformation that would introduce a loop in the tree.

4 Performance Evaluation

We implemented ACDC in the ns network simulator. For most of our experiments, we use 600-node transit-stub GT-ITM topologies with 120 overlay participants.

ACDC achieves the target delay as long as it is at least 5% larger than the delay of a degree-unbounded shortest path tree. Similarly, the cost of the overlay comes within 13% of the MST cost when the delay bound is relaxed sufficiently. Per-node probing overhead is independent of the number of overlay participants, rising to less than 900 bytes/sec/node for a probe set size of 20. With a probe set size of 10 (2440-node topology), it takes approximately 200 seconds to bring 95% of participants within the delay bound of 1.2SPT. ACDC is resilient to highly dynamic network conditions (substantial link delay increases and node failures).

Encoded Meta-Data Formats for Internet Subscription Systems

Joanna Kulik*
MIT Laboratory for Computer Science
jokulik@lcs.mit.edu
http://mimas.lcs.mit.edu/~jokulik/event.html

Our work addresses the design of Internet Subscription Systems, a general category of communication system currently under development that encompasses both event-notification and publish-subscribe systems. Put simply, Internet Subscription Systems are mechanisms for notifying subscribers as quickly as possible to the arrival of relevant information on the Internet. Subscribers first sign up for notifications on topics that are important to them. As soon as relevant information on a topic arrives on the Internet, it is routed to the appropriate subscribers. This information can include news alerts, auction information, stock market quotes, weather reports, and many other notices. Subscription systems can also support large-scale interactive games and transmission of online entertainment events.

Because Internet Subscription Systems have the potential to benefit so many applications, researchers have proposed a panoply of approaches for implementing such systems. As these systems have evolved, each one has added features that a previous system lacked. IP multicast systems improve upon unicast systems because they can quickly distribute notifications to large numbers of subscribers without overwhelming the network. Content-based routing systems improve upon IP multicast systems by providing subscribers with a large, descriptive namespace for identifying notifications. As each system adds new features, however, it likewise adds an increasing amount of complexity to routers in the network. These systems may also sacrifice some features in favor of adding new ones. For example, content-based routers read each message's contents, potentially compromising the privacy of users in a way that IP multicast routers do not.

The main purpose of our work is to construct a system that outperforms those in use currently. We propose a general approach to designing such systems which relies on the use of encoded, meta-data headers. In this approach, encoders would first extract features from message contents that are relevant to routing. They would then translate these meta-data features into an encoding format specified by the subscription system and attach the encoding to each message's header. The subscription system would, in turn, route the message based upon this encoded header only. The challenge is for researchers to come up with an encoding format that captures the complexity of real-world applications but is designed specifically for efficient processing by routers.

We have designed a specific subscription system using this approach, The Graph-Encoded Name Notification Architecture, or GENNA. GENNA encoders extract information from messages and encode them into a format called a content graph. The design of content graphs rests upon a single observation. In order to route messages, routers need to be able to detect only *the relationships between messages*, not information about the messages themselves. Forcing a router to weed through this information merely slows it down while compromising the privacy of its users. Content graphs therefore reveal message relationships in terms of a partially-ordered digraph. GENNA routers use this graph information to set up subscriptions and match them with incoming notifications.

We have performed a series of simulation experiments in order to compare content graphs against other encoding formats. In these experiments, we applied different encoding formats, including content graphs, to a variety of real-world applications, including baseball, stock-market, airport, and traffic announcement systems. Our results show that systems that disseminate notifications using IP multicast headers performed the least efficiently of all the systems that we studied. Two of the header formats that we studied, including content graphs, performed significantly better than IP multicast. The key advantage of content graphs is that they help reduce the size of notification headers and therefore the overhead of disseminating notifications.

*This effort was sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-00-2-0553.

A Lightweight, Robust P2P System to Handle Flash Crowds

Angelos Stavrou* Dan Rubenstein* Sambit Sahu†
*Dept. of Electrical Engineering, Columbia University, New York, NY
†IBM T.J. Watson Research Center, Hawthorne, NY
{angelos,darn}@ee.columbia.edu sambits@us.ibm.com
URL: http://www.comet.columbia.edu/~angelos

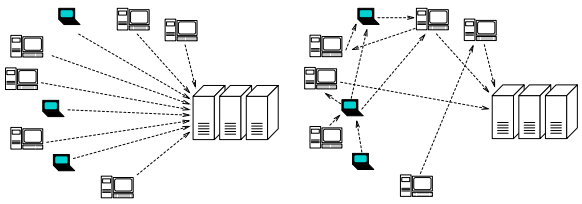
Introduction

Internet flash crowds (a.k.a. hot spots) are a phenomenon that result from a sudden, unpredicted increase in an on-line object's popularity. Currently, there is no efficient means within the Internet to scalably deliver web objects under hot spot conditions to all clients that desire the object.

Proposed Solution

We present a system that forms a peer-to-peer (P2P) overlay network that allows clients that have received copies of the popular content to forward the content to those clients that desire but have not yet received it. We work on top of overlay topologies generated essentially at random to scalably, reliably deliver content whose popularity exceeds the capabilities of standard delivery techniques.
We call this system PROOFS:
***P**2P **R**andomized **O**verlays to **O**bviate **F**lash-crowd **S**ymptoms*.

Application of PROOFS in a network



(a) "Standard" architecture *(b) PROOFS-enhanced architecture*

Design Description

PROOFS consists of two protocols. The first forms and maintains a network overlay that connects the clients that participate in the P2P protocol. The overlay construction is an ongoing, randomized process in which nodes continually exchange neighbors with one another at a slow rate. The second protocol performs a series of randomized, scoped searches for objects atop the overlay formed by the first protocol. Objects are located by randomly visiting sets of neighbors until a node is reached that contains the object.

Advantages over DHT-based approaches (e.g., CAN,Chord,Pastry,Tapestry)

- Clients are not required to cache any objects or pointers to objects other than that which the client has explicitly expressed interest in receiving.
- PROOFS handles dynamic changes in overlay membership (i.e., participants joining and leaving) without any additional mechanism or modification to its fundamental design.
- PROOFS is naturally robust even when there exist a substantial number of clients who "take advantage" of the system by using it to obtain popular objects, but who do not fully participate in assisting other clients by either refusing to forward content or even secretly dropping all queries it receives.
- The system is amenable to the formation of complex queries that contain keywords or temporal restrictions (e.g., an object generated within the last 5 minutes).

Analytical Results

Through analytical modeling and simulation, we show that the likelihood that the constructed overlay separates a client from reaching a large fraction of other clients is extremely rare, even in the presence of clients dynamically joining and leaving the overlay. Through simulation, we show that traffic levels, latency, and connectivity grow in a tolerable manner as a function of the fraction of overlay nodes cease to perform query and object forwarding (i.e., non-cooperative nodes).

Experimental Results

We evaluate a prototype implementation on a testbed comprised of end-systems scattered around the world. Although small in scale compared to how we hope the system will eventually be used, the testbed demonstrates that latencies and traffic utilizations by the system are low enough to make the approach feasible in today's networks.

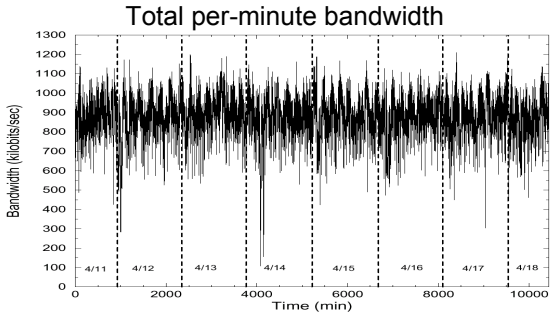
*This material was supported in part by the National Science Foundation under Grant No. ANI-0117738. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Provisioning On-line Games: A Traffic Analysis of a Busy Counter-Strike Server

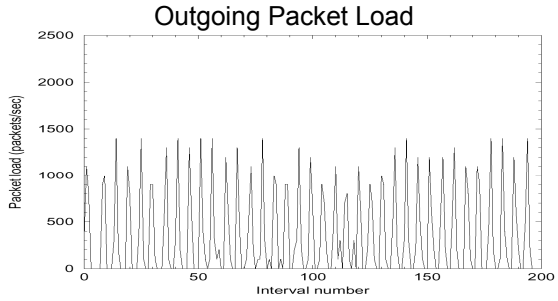
Francis Chang, Wu-chang Feng, Wu-chi Feng, Jonathan Walpole
{francis, wuchang, wuchi, walpole}@cse.ogi.edu
OGI School of Science and Engineering at OHSU

Abstract:

This poster describes the results of a 500 million packet trace of a popular on-line, multi-player, game server. The results show that the traffic behavior of this heavily loaded game server is highly predictable and can be attributed to the fact that current game designs target the saturation of the narrowest, last-mile link (i.e. 56k modems). Specifically, in order to maximize the interactivity of the game itself and to provide relatively uniform experiences between players playing over different network speeds, on-line games typically fix their usage requirements in such a way as to saturate the network link of their lowest speed players. While the traffic observed is highly predictable, the trace also indicates that these on-line games provide significant challenges to current network infrastructure. As a result of synchronous game logic requiring an extreme amount of interactivity, a close look at the trace reveals the presence of large, highly periodic, bursts of small packets. With such stringent demands on interactivity, routers must be designed with enough capacity to quickly route such bursts without delay. As current routers are designed for bulk data transfers with larger packets, a significant, concentrated deployment of on-line game servers will have the potential for overwhelming current networking equipment.



The trace data that we observed does not conform to traditional models of network traffic. Specifically, the traffic does not exhibit self-similar, or heavy-tailed properties. The long-term behavior is characterized by a near-constant consumption of network resources. However, on smaller scales, the traffic is dominated by an extremely bursty, highly periodic pattern.



The real-time low-latency requirements of the application's logic places demands on network infrastructure which are dissimilar from more traditional Internet services. Thus, network packets are relatively small – packet data is often smaller than the link and transport layer headers. In addition, packets are broadcast on intervals much shorter than typical retransmission times, UDP is typically chosen to satisfy this archetype of networking requirements.

With the global explosion of on-line multi-player gaming, it will become more important in the coming years to understand how the Internet environment will be influenced by this form of traffic.

[1] W. Feng, F. Chang, W. Feng, J. Walpole, "Provisioning On-Line Games: A Traffic Analysis of a Busy Counter-Strike Server", In Proceedings of the Internet Measurement Workshop, November 2002
http://www.cse.ogi.edu/sysl/projects/cstrike/

This material is based upon work supported by the National Science Foundation under Grant EIA-0130344 and the generous donations of Intel Corporation. Any opinions, findings, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Intel.

Adaptive Bandwidth Control for Efficient Aggregate QoS Provisioning*

Peerapon Siripongwutikorn† Sujata Banerjee†‡ David Tipper†

In recent years, aggregate class-based traffic management and control have been used to mitigate scalability issues in backbone networks. The principle of aggregate based traffic control is embodied in Virtual Private Network (VPN) applications and the Differentiated Service (DiffServ) framework. However, attempting to provide statistical quality of service (QoS) guarantees for the aggregate traffic classes is difficult due to the unpredictable temporal and spatial variations in the aggregate traffic. Under these circumstances, using static bandwidth allocation approaches almost always ends up overprovisioning bandwidth. This not only leads to poor network utilization but also penalizes best-effort traffic whose performance hinges on the unused available bandwidth left over from QoS traffic.

The motivation for this research is the need to guarantee the aggregate QoS while being able to attain efficient network bandwidth utilization. This work presents an Adaptive Bandwidth Control (ABC) approach based on fuzzy control to maintain the aggregate loss QoS in a general queueing system with unknown statistical characteristics of the input traffic. The control mechanism involves periodic bandwidth adaptation to ensure that the allocated bandwidth is just enough to maintain the specified QoS requirement. As such, less bandwidth will be wasted due to overallocation. The appealing aspects of our work are that (i) the quantitative QoS guarantee with high assurance under dynamic traffic conditions is achieved, (ii) only the aggregate average rate information (which can be measured on-line) is required for the control, and (iii) the loss QoS metric is guaranteed over both short and long time scales. Maintaining short term QoS is very important for audio and video applications and relatively little work exists on maintaining short term QoS in a bandwidth efficient manner. Our ABC algorithm is also computationally simple and thus incurs minimal processing overhead.

Our proposed ABC algorithm to maintain the packet loss at a desired target value belongs to the class of feedback control algorithms in which the allocated bandwidth is adjusted based on some system state feedback. However, using packet loss as feedback has some fundamental drawbacks including its susceptibility to large measurement errors and the need for a long measurement period. Instead we map the target loss into two target queue thresholds with the control objective of maintaining the average queue length between these two tight thresholds. Currently, we use a simple mapping based on an $M/M/1/K$ queueing model.

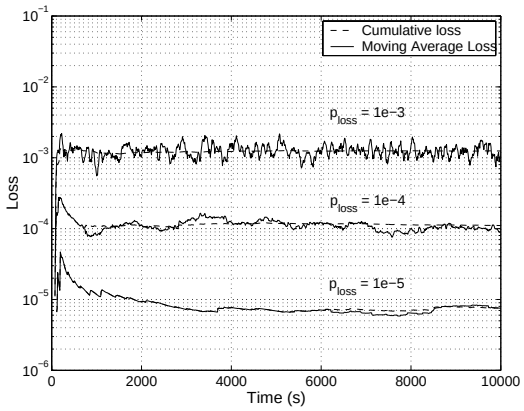


Figure 1: Loss in a queue with non-stationary LRD traffic

Results from a detailed simulation study demonstrate that the proposed control algorithm can accurately maintain short-term packet loss under non-stationary traffic conditions, where flows in the aggregate arrive and depart over time, and with long range dependent (LRD) traffic consisting of Pareto on-off sources (Fig. 1). Furthermore, the control mechanism yields high link utilization and is robust over a wide range of target loss values and control parameters. With a specific stationary traffic scenario, the average bandwidth allocated by the proposed control is only 2-3% higher from the optimal static bandwidth allocation (which is a lower bound determined by trial-and-error and post-analysis of the entire simulation run and not practically achievable). With a specific non-stationary traffic scenario, the bandwidth savings from using the proposed control is approximately 9% with the coefficient of variation of the short term loss being 4-8 times lower, as compared to the static optimal allocation. Further, using an equivalent bandwidth formula for LRD traffic, where the bandwidth is reallocated at the arrival and departure instants of each flow, we found that the average allocated bandwidth is 35% higher than that required by the fuzzy control mechanism. Note that keeping track of per-flow information such as the arrival and departure instants, hinders scalability and is not likely to be implemented in most aggregate traffic handling systems.

In our research thus far, the proposed ABC control mechanism has been studied in a single queue system. More sophisticated and accurate packet loss/queue thresholds mappings, appropriate (and perhaps adaptive) choice of the control time scale, as well as the control of multiple queues are being investigated in the on-going research.

*Supported in part by the NSF CAREER award No. ANI-9702389.

†Telecommunication Program, University of Pittsburgh

‡Hewlett-Packard Laboratories

§Email: {*peerapon,sujata,dtipper*}*@mail.sis.pitt.edu*. A longer version of this work is available at `http://www2.sis.pitt.edu/~peerapon/papers/report.pdf`.

Oblivious AQM and Nash Equilibria *

Debojyoti Dutta, Ashish Goel, John Heidemann
ddutta@isi.edu, agoel@usc.edu, johnh@isi.edu
http://www.isi.edu/saman/game/

An oblivious Active Queue Management (AQM) scheme is one that does not differentiate between packets belonging to different flows. For example, Fair Queueing is not oblivious, while RED is. The current Internet is dominated by TCP traffic. However, there are indications that the amount of non-congestion-reactive traffic is on the rise. TCP (and in fact, any transport protocol that we are aware of) does not guarantee good performance in the face of aggressive, greedy users who are willing to violate the protocol to obtain better performance. Thus, it seems important to study scenarios where end-points are greedy and selfish, and do not follow socially accepted congestion control mechanisms.

Selfish users can be modeled using game theory. A game consists of rules and players. In the Internet game, the rules are set by the AQM policies, and the players are the end-point selfish traffic agents. A fundamental solution concept in game theory is the Nash equilibrium. In the context of our problem, a Nash equilibrium is a scenario where no selfish agent has any incentive to deviate from its current state to increase its own utility. Thus, the existence of a Nash equilibrium implies a stable state of the network in the presence of selfish users, but does not provide any clues as to how this state should be achieved.

In this work, we study the existence and the properties of Nash equilibria imposed by oblivious AQM schemes on selfish agents, which generate Poisson traffic but can control the average rate. We will restrict ourselves to AQM strategies that guarantee bounded average buffer occupancy, regardless of the total arrival rate. Oblivious AQM schemes are of obvious importance because of the ease of implementation and deployment, and Nash equilibrium offers valuable clues into network performance under non-cooperative user behavior.

Specifically, we ask the following three questions:

1. **Existence**: Are there oblivious AQM schemes that impose Nash equilibria on selfish users?
2. **Efficiency**: If an oblivious AQM scheme can impose a Nash equilibrium, is that equilibrium good or *efficient*, in terms of achieving high goodput and low drop probability?
3. **Reachability/Achievability**: How can the users / players reach the equilibrium point, if one exists?

We first derive a necessary and sufficient condition, that we call the Nash condition, for oblivious AQM schemes to impose Nash equilibria on selfish users. We show that the popular oblivious AQM strategies, drop-tail and RED, do not impose Nash equilibria on selfish users. Then we propose a variation of RED, *VLRED*, that can impose a Nash equilibrium but the utilization at the equilibrium point drops to 0 asymptotically as the number of users increases. This motivates us to develop another AQM scheme, Efficient Nash AQM (EN-AQM), which can impose a Nash equilibrium, and guarantee strict bounds on equilibrium performance; that is, provide lower bounds on goodput, bounded average delays, and upper bounds on drop probability at equilibrium. It is surprising that oblivious schemes can have such strong properties. The details can be found in a technical report [1].

We also observe that for any oblivious AQM scheme, the Nash equilibrium imposed on selfish agents is highly sensitive to the increase in the number of users, making it hard to deploy and difficult for users to converge to. This further motivates the need for protocols which lead to an efficient utilization and a somewhat fair distribution of network resources (like TCP does), and also ensure that no user can obtain better performance by deviating from the protocol. We use the term *protocol equilibrium* to describe this phenomenon. If protocol equilibrium is achievable, then it would be a useful tool in designing robust networks. This appears to be an interesting, and an open problem.

* The authors are at the Computer Science Department and the Information Sciences Institute at the University of Southern California. This work is supported, in part, by DARPA and the Space and Naval Warfare Systems Center San Diego (SPAWAR) under Contract No. N66001-00-C-8066. Ashish Goel's research was also supported, in part, by NSF CAREER Award no. 0133968.

References

[1] Debojyoti Dutta, Ashish Goel, and John Heidemann. Oblivious AQM and Nash Equilibria. Technical report, USC-CS-TR-763, July 2002.

A Comparison of RED's Byte and Packet Modes*

Wesley M. Eddy, Ohio University, weddy@irg.cs.ohiou.edu
Mark Allman, BBN Technologies/NASA GRC, mallman@bbn.com

Longer version: http://irg.cs.ohiou.edu/~weddy/papers/redmodes.ps

The Random Early Detection (RED) active queue management scheme [FJ93], recommended for deployment by the IETF [B98], is used by routers to control their congestion levels by informing end hosts of incipit congestion. Provisions in the original RED research give routers the option of measuring their queue length in terms of the number of packets buffered or in terms of the number of bytes of buffer space consumed by the queued packets. In addition, a packet's size may factor into the computation of its *marking probability* (the probability a given packet will be dropped or the end host notified via ECN to reduce the sending rate). Given a network with heterogenous packet sizes (e.g., the Internet) the choice of using packets or bytes as the units in RED's calculations can have implications on performance and fairness. As an example, when queueing and marking in terms of packets a RED queue may mark small packets at an inappropriately high rate based on the small packet's use of the bandwidth. Through simulations we quantify the performance implications of selecting amongst combinations of queue measurement and marking modes from the standpoints of link utilization and fairness. Additionally we have investigated the impact of setting RED's *mean packet size* parameter that the byte modes depend upon.

Simulations of a link congested with several bulk FTP transfers show that selecting packet modes provides greater link utilization, while byte modes enhance fairness among competing flows. We also observe that the decision of whether or not to adjust the marking probability of a packet based on its size has more effect on performance than the choice of units for queue measurement. These observations hold constant across different mixes of packet sizes and traffic levels, however the results are most apparent when there is a high load and wide range of packet sizes.

Our next set of simulations involves HTTP traffic congesting the RED gateway. We first note that in lightly loaded networks all RED variants performed roughly the same and therefore choosing one particular variant does not seem important. In networks with moderate to heavy congestion levels we note that marking based on a packet's size yields higher fairness. Also, measuring the queue in terms of bytes causes better link utilization in our simulations, while at the same time lowering the average queueing delay and reducing the degree of jitter in queueing delay. Since the majority of real-world Internet traffic today consists of such HTTP transfers, this makes a strong argument for consideration of byte-based RED modes for deployment.

RED's byte modes use a constant called the *mean packet size* (MPS) to calculate a given packet's probability of being marked. Next, we investigate how difficult it may be to set the MPS and the sensitivity of our results to the MPS setting. We examine packet traces collected at two different points in the Internet. Our analysis indicates that the degree of mixture and rate of change in packet sizes varies across both time and network. We conclude from this that the MPS must remain a site-selectable parameter – or possibly something that RED could tune on-the-fly. Our simulations of both FTP and HTTP traffic show that the MPS parameter should be *approximately accurate* for the mix of traffic traversing a link, but does not need to be exact. Specifically, we note that settings within several hundred bytes of the measured mean packet size are roughly equivalent in performance.

Finally, we introduce a rating system for comparing the performance of various RED modes based on link utilization and fairness, and favoring some degree of balance between the two metrics. We show that the combination of a queue measured in bytes and marking based on the number of bytes in a packet generally offers the best ranking among other options tested across our simulation scenarios. However, a concrete suggestion on which RED variant to use remains elusive, as in some scenarios byte modes rank lowest. Therefore, we conclude that some testing of different modes may be necessary on a site-by-site basis in order to achieve desired performance properties.

*The authors thank NASA's Glenn Research Center for funding this work.

References

[B98] Robert Braden et. el. Recommendations on Queue Management and Congestion Avoidance in the Internet, April 1998. RFC 2309.

[FJ93] Sally Floyd and Van Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

Robust Rate based Congestion Control

Ganesh Gopalakrishnan {ggopalak@uiuc.edu}
Sneha K. Kasera {kasera@research.bell-labs.com}

1 Introduction

Routers need to implement congestion control measures to overcome the problem of reduced performance in the event of congestion. In this work, with a view towards robust implementation, we examine a new active queue management scheme, Acceptance and Departure Rate (ADR) that uses a combination of acceptance and departure rate measures to control link congestion in a router. Unlike many existing approaches, ADR does not use queue length measures which are not robust to changes in system capacity. The main advantage of our approach is the robustness of the system with varying network parameters including link capacity and network load.

2 Our Proposal

We use the acceptance and departure rates (ADR) to compute the fraction of input traffic that can be allowed into the system. The acceptance rate, defined as the number of bits accepted in the queue in a given time interval, provides a measure of offered load. The departure rate, defined as the number of bits transmitted on the link, a measure of processed load. A combination of both is essential in providing fast but stable response under different load conditions. Control based on the processed load provides stability and backlog clearance whereas the control based on offered load provides fast response.

In our timer-based ADR scheme, the system load (in terms of acceptance and departure rates) is measured periodically, based on which a *"fraction allowed"* (the fraction of the input traffic that can be allowed into the router queue) is computed. This fraction allowed is then used to drop packets in the next interval. In a timer-based scheme the algorithm to compute the fraction allowed need not be executed in the fast path. Only the actual packet drop decision based on the fraction allowed need be taken in the fast path. In addition to saving processing overhead in the fast path, a timer-based scheme gives the freedom to choose different and even complex policies for determining the fraction allowed without modifying the packet forwarding hardware.

In our algorithm, we periodically measure acceptance and departure rate and then normalize them with respect to line capacity. Next we compare the normalized acceptance rate with a threshold and determine the fraction allowed based on acceptance rate measure only (f_a). We also compare the normalized departure rate with another threshold and determine the fraction allowed based on departure rate measure only (f_d). The actual fraction of traffic allowed into system is chosen to be $f = \min(f_a, f_d)$. Furthermore, in order to prevent oscillations, the fraction allowed in an interval is smoothed by multiplying f with the fraction allowed in the previous interval. Once the fraction allowed is computed, we use a deterministic algorithm [Hajek 1985] for dropping packets. This deterministic algorithm demonstrates much less variability from the desired fraction allowed in comparison to other probabilistic drop mechanisms [Kasera 2001]. The randomness in packet arrivals ensures that no particular source is able to misuse the deterministic nature of the algorithm.

The timer-based ADR requires specification or determination of two timers, one associated with acceptance rate measurements and another with departure rate measurements. It also requires specification of the normalized acceptance rate and departure rate thresholds. In order to fully utilize the link, the departure rate threshold should be set close to 1. The acceptance rate threshold could be set to higher than one to allow some queuing during short bursts. The acceptance rate timer should be chosen to be less than the departure rate timer to allow the acceptance rate control to react faster to sudden bursts of load. The departure rate control should be applied over larger timer intervals. Both the timers should be set to less than the maximum queuing delay experienced in the router. The maximum queuing delay could be part of the router specification or measured from the expression B = C*D where B is the buffer size, C is the line capacity and D is the maximum queuing delay.

We use ns-2 simulator to evaluate the performance of ADR. In particular, we examine Link Utilization, Goodput, Loss and Delay. We have conducted extensive tests with varying network parameters and also compared ADR's performance with other AQM schemes. Our results show that ADR is fair and robust against varying system and network parameters like link Capacity, buffer size, network load as well as with different traffic types like TCP and UDP.

For a full description, analysis of ADR, visit http://www.uiuc.edu/∼ggopalak/

Using SCTP Multihoming for Fault Tolerance and Load Balancing*

Armando L. Caro Jr. and Janardhan R. Iyengar
Paul D. Amer, Gerard J. Heinz, Randall R. Stewart†

http://pel.cis.udel.edu

Protocol Engineering Lab
CIS Department, University of Delaware
{acaro, iyengar, amer, heinz}@cis.udel.edu

† Cisco Systems Inc.
rrs@cisco.com

1 SCTP Overview

Mission critical systems rely on redundancy at multiple levels to provide uninterrupted service during resource failures. Such systems when connected to IP networks often deliver network redundancy by multihoming their hosts. A host is multihomed if it can be addressed by multiple IP addresses. An endpoint's IP address can become inaccessible, possibly due to an interface failure, severe congestion, or due to BGP's slow route convergence around path outages. Redundancy at the network layer allows a host to be accessible even if one of its IP addresses becomes unreachable; packets can be rerouted to one of its alternate IP addresses.

TCP does not support multihoming. Any time either endpoint's IP address becomes unreachable, TCP's connection will timeout and abort, thus forcing the upper layer to recover. The recovery delay can be unacceptable for mission critical applications such as IP telephony, IP storage, and military battlefield communications. To address TCP's shortcoming, the Stream Control Transmission Protocol (SCTP) has been designed with fault tolerance in mind. SCTP supports multihoming at the transport layer to allow SCTP associations to remain alive even when an endpoint's IP address becomes unreachable.

2 Adaptive Failover Mechanism

SCTP has a built-in failure detection and recovery system, known as failover, which allows associations to dynamically send traffic to an alternate peer IP address when needed. SCTP's failover mechanism is static and does not adapt to application requirements or network conditions.

Network dynamics however, vary greatly among different networks and hence, the heuristics used for determining failover should be adjusted accordingly. Applications also have different requirements that the failover mechanism should cater to. For example, SS7 signalling applications require that failovers take no longer than 800 ms. On the other hand, a file transfer may be more concerned with the total transfer time. Therefore, we argue that network fault tolerance should cope with dynamic network conditions and varying application needs.

We have developed a two-level (alpha-beta) threshold mechanism for SCTP which provides added control over failover actions. We have formally specified and modeled our failover mechanism, and are currently investigating the relationships between failover thresholds and network parameters, such as round trip times, packet loss rates, etc. From these relationships, we will develop an adaptive failover mechanism for SCTP.

3 End-to-End Load Balancing

SCTP provides for application-initiated changeovers so that the sending application can change the sender's primary destination address, thus moving the outgoing traffic to a potentially different path. Although the motivations for providing a changeover mechanism are different, it is not difficult to envision application developers using this feature for load balancing at the application layer.

With the provisioning for multihoming in SCTP, we believe that end-to-end load balancing can be performed at the transport layer. Being better informed than the application layer about the end-to-end paths, the transport layer can perform fine-grain load balancing. We foresee issues during load balancing in areas such as congestion control and loss detection and recovery. These issues also suggest that the transport layer be involved.

We are currently looking at issues due to changeover. We have uncovered a problem that results in cwnd overgrowth during changeover. Analysis shows that this problem may not be a corner case but may occur under various network and changeover conditions. We propose the Rhein algorithm and the Changeover Aware Congestion Control (CACC) Algorithms as solutions to the problem. In the future, we will investigate (1) end-to-end techniques for shared bottleneck detection to aid in performing correct congestion control during load balancing, and (2) algorithms for scheduling traffic on the multiple paths.

* Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

AFBV: A Scalable Packet Classification Algorithm

Ji Li, Haiyang Liu, Karen Sollins
MIT Laboratory for Computer Science
{jli, hyliu, sollins}@ mit.edu
http://www.mit.edu/~jli/afbv

Packet classification is a central function of a number of network activities, such as routing and firewall management. The packet classifier has a ordered rule database, one rule for each type of packets, and all arriving packets are classified into equivalent classes based on the first rule they match. With the development of the Internet, the database size can only grow with time. Consequently, scalability becomes an important issue for packet classification algorithm. This project is addressing the scalability problem in the selection of rules for packet classification.

There is significant previous work in this area. Much previous work executed in linear time with the number of rules. Our work is an improvement on the ABV scheme of Baboescu and Varghese [1], which in turn is an improvement on the straightforward linear Lucent bit vector scheme (BV) [2]. The BV scheme produces bit vectors of all rules for each field, and selects the rules applicable to the complete header by first finding the bit vectors corresponding to each field, then intersecting them, and the first set bit in the consequent bit vector implies the position of the rule in the database. ABV made two observations: first the set bits in the bit vectors are sparse and second a packet matches no more than a few rules. Hence, ABV improves BV to achieve logarithmic time in two steps. First, ABV rearranges the rules by sorting, so that those rules that match specific prefixes are near each other in the list. Second, it aggregates bits in each field to reduce memory accesses, but increasing the possibility of false matches. The rule rearrangement mitigates the false matches to some extent, but also incurs additional cost. First, rule rearrangement makes it necessary to find all matches, instead of first match in BV. Second, it is not easy to find an effective rearrangement method. Third, rules need to be mapped back to the original order so as to find the rule with the lowest order.

Our scheme, the Aggregated and Folded Bit Vector algorithm (AFBV) inherits the idea of bit vector and aggregation from BV and ABV, but discards rule rearrangement and introduce a new concept: folding. All bits in a predefined range r (r is called *folding range*, usually equal to multiple aggregate sizes), whose positions have the same mod-f value are OR-ed together, the result occupying that position mod f, thus folding the bit vector (f is called *folding size*). Only when there are at least one bit set at the same mod-f position can one get false matches, while in ABV, a match among any bits within an aggregation group can lead to a false match. Once there is a match in the folded bit vector at position k (k is inclusively between 0 and f-1), the possible real match position must take the form ($I \cdot f + k$). Therefore, folding helps not only filter out false matches but also locate the real match positions. This scheme will not produce any false negatives. Thus, correctness is guaranteed.

Besides the folding range and the folding size, the number of folding is another important parameter in AFBV. By using multiple folding with different sizes, we can enhance the detecting and locating ability of the folded vectors. First, only if there are matches in all the intersections of the folded vectors may there be real matches in the original vectors. The gain here is that mostly we do not need to fetch the whole original bit vector to conclude a false match as required in BV and ABV. Second, with proper selection on the folding sizes, multiple folding greatly reduces the number of the possible match positions than single folding. The gain here is that in case of real matches, we only need to fetch parts instead of the whole original bit vectors.

In addition to the aggregate bit vectors, AFBV needs to store the folded bit vectors, which incurs about 10% additional storage requirements in our case.

AFBV offers the following benefits: Only the first matched rule need to be found; no mapping back cost is needed; without rule rearrangement, pre-processing is much simple; a large aggregate size can be applied because the folded vectors help to reduce false matches.

In our simulations of relative performance we compare the Lucent BV algorithm, the ABV algorithm and our AFBV algorithm, and evaluate both performance and space utilization. Preliminary results on different-sized databases show that AFBV outperforms ABF and BV on average, and can achieve the similar level of performance in worst cases, with about a 10% increase in memory requirements, for the additional, folded vectors.

* This work was funded by the National Science Foundation under grant number CCR-0122419.

References:

1. F. Baboescu, G. Varghese. Scalable Packet Classification. In Proc. ACM SIGCOMM'01, Aug. 2001.
2. T. Lakshman and D. Stidialis. High speed policy-based packet forwarding using efficient multi-dimensional range matching. In Proc. ACM SIGCOMM'98, Sept. 1998.

Anypoint: Extensible Transport Switching on the Edge

Ken Yocum, Darrell Anderson, Jeff Chase and Amin Vahdat *
Department of Computer Science, Duke University
{grant,anderson,chase,vahdat}@cs.duke.edu

Anypoint is a new model for one-to-many communication with ensemble sites—aggregations of end nodes that appear to the external Internet as a unified site. Policies for routing Anypoint traffic are defined by application-layer plugins residing in extensible routers at the ensemble edge. Anypoint differs from previous indirection approaches in that its switching functions operate at the transport layer, at the granularity of transport frames. Anypoint *transport switching* preserves end-to-end transport behavior for rate control, ordering, and reliable delivery. As in illustrating example, we have built an NFS storage router using a host-based Anypoint prototype.

Anypoint generalizes "L4-L7" server switches, commonly used for server load balancing. The Anypoint architecture supports fine-grained indirection for content-based request routing, resource management, and response merging. Anypoint allows independent, concurrent handling of multiple requests arriving on the same persistent transport connection, enabling a new class of virtualization switches for network storage protocols and other Internet services. In contrast, commercial Web switches are limited to service protocols (e.g., HTTP) and issue each request in a separate transport connection, or process requests on each connection serially.

Anypoint enables transparent extension and virtualization of network services based on application-layer (L7) protocols. Indirection at the intermediary hides the ensemble's internal structure from the connection peer. Clients send inbound traffic to an Anypoint switch using a *virtual IP address* (VIP) for the ensemble site. The switch merges outbound traffic for the peer into a single stream with the VIP as the source address. Service-specific *Application-Layer Routing Modules* (ALRMs) in an Anypoint switch examine, transform, and redirect traffic. In this respect, Anypoint is in the spirit of Active Networks and subsequent proposals for enhanced, extensible router architectures.

A key principle of our approach is that transport functions such as buffering, reassembly, acknowledgment, and retransmission are handled in an end-to-end fashion by the end nodes. Anypoint communication is reliable, partially ordered, and rate-controlled and complements IP-layer approaches (e.g., Anycast and i3), which support wide-area ensembles but provide only best-effort packet delivery. Note that—in contrast to application-level proxies—an Anypoint intermediary does not terminate transport connections.

Anypoint is the first general indirection approach for connection-oriented transports that operates at the granularity of frames. Anypoint is designed for advanced IP transports with partial ordering and application-level framing, as proposed by Clark and Tennenhouse over a decade ago. This approach is more powerful and elegant than solutions based on TCP connection migration. These transport capabilities help limit data buffering to port queues and frame reassembly, making our approach amenable to implementation on high-speed switches and routers.

The Anypoint prototype uses a nonstandard IP-based transport protocol that is framed and partially ordered. We reuse code from TCP, adding framing support based on a subset of the TCP upper-layer framing (TUF) proposal that we call TUF-lite. Each TUF-lite segment is self-describing and contains an integral number of complete frames. A small amount of code at the socket layer supports TUF sockets with UDP-like messaging semantics (*sendto, recvfrom*). Anypoint is *transport equivalent*: end nodes use the same transport code for point-to-point and Anypoint connections. We believe that the Anypoint model is fully compatible with SCTP, an accepted IP transport with framing. The switch is implemented as a loadable FreeBSD kernel module.

The contributions of this research are to (1) show that transport frame switching at the network edge is a powerful technique to virtualize and extend Internet services, (2) define an extensible framework and mechanisms that enable this processing, and (3) explore the implications for Internet service structure, extensible routers, and IP transport protocols.

*This work is supported in part by the U.S. National Science Foundation (CCR-00-82912, EIA-9972879, and EIA-9870728), Network Appliance, and Cisco Systems, and by an equipment grant from IBM.

The Stanford Virtual Router

A teaching tool and network simulator

Martin Casado casado@stanford.edu
Vikram Vijayaraghavan vvikram@cs.stanford.edu
Guido Appenzeller appenz@cs.stanford.edu
Nick McKeown nickm@stanford.edu

Department of Computer Science, Stanford University
Contact: appenz@cs.stanford.edu 650-723-1414

The virtual router project is an effort to develop a platform to facilitate research and teaching of network routing. It allows to set up a virtual topology of routers and network links and to run user-space programs on the simulated hardware. Unlike existing network simulators such as ns2, the system operates with real IP packets in real time. This makes it possible for students or researchers to generate traffic with real, standard clients and servers and evaluate performance over the simulated topology.

In virtually all modern operating systems, routing is implemented within kernel space. Providing a good kernel debugging environment and assuming a basic knowledge of kernel level development may be restrictive requirements for a project focused on routing. Take for example an undergraduate research course, not only would the previous two requirements have to be met, but students could only work one to a computer. For these reasons, routing typically has been taught either from a theoretical perspective or through the use of a software simulator such as ns2. The virtual router project provides full access to actual network traffic, but bypasses the trouble of having to work within the kernel.

Some experiments such (e.g. routing protocols such as RIP) require more than a single router. The virtual router makes it possible to simulate large number of routers on one physical host and enforce a virtual network topology between them. This simulated topology can be connected to real networks. Packets can travel from a physical network (e.g. the internet) through the simulated topology and back to the physical network.

The Virtual Router system can be broken down into two components, the virtual router server, and a number virtual router clients. The virtual router server is an application or set of applications running on a physical server that have access to link layer traffic. The virtual router client is a program created by users of a VR system. In a typical session, a user will use a VR client to connect to the server via a TCP socket, and a decision is made about what packets the server will send to the client. The user may then run a network application, such as FTP, whose traffic can be seen by the server. The server, on capturing traffic, will based on the simulated topology decide whether or not a particular client can see the packet. If this is the case it is forwarded to the client who can then manipulate the packet (e.g. decrement the TTL field). The client may then send the packet back to the server with instructions to send it out of a particular interface on the network, thus potentially routing the packet. With its relationship with the server, the VR client has full capability to manipulate and route traffic flows from user space.

The virtual router has successfully been used to teach students basic functionality of routers. Students had to implement virtual router clients that make routing decisions, use the ARP protocol to communicate with other routers and route actual web traffic from their desktop computers. However the scalability of the virtual router (up to several hundred simulated routers) makes it attractive as a research tool. Currently there are efforts under way to investigate the behavior of routing protocols using the virtual router as an experimental platform.

Netbed: An Integrated Experimental Environment

Brian White, Shashi Guruprasad, Mac Newbold, Jay Lepreau
Leigh Stoller, Robert Ricci, Chad Barb, Mike Hibler, Abhijeet Joglekar

School of Computing, University of Utah

www.cs.utah.edu/flux www.emulab.net www.netbed.org

The diverse requirements of network and distributed systems research are not well met by any single experimental environment. Competing approaches remain popular because each offers different levels of *ease of use*, *control*, and *realism*. These include packet-level discrete event simulation, live network experimentation, and emulation, which subjects real hardware, protocols, and workloads to a synthetic network environment.

Netbed offers a new alternative. It is software that, when deployed on local and wide-area machines, provides a platform for research, education, or development in distributed systems and networks. Netbed's primary contribution is the seamless *integration* of the above disparate techniques into a common architectural framework that preserves the control and ease of use of simulation, without sacrificing the realism of emulation and live network experimentation. This framework provides common abstractions, namespaces, services, and user interfaces to all three environments. Netbed's integration allows tools such as topology and traffic generators that were originally targeted for one domain to apply to all.

Netbed leverages and extends its ancestor, *Emulab*, which focused on making *emulation* as easy to use and control as simulation. Our results show that Emulab speeds up experiment configuration on a cluster by two orders of magnitude, and through time- and space-sharing, improves cluster utilization by another two orders of magnitude.

Architecture: Netbed revolves around interacting state machines, monitored by a state management daemon. The central state machine is the "experiment." Subsidiary state machines include those handling node allocation, configuration, and disk reloading. This approach copes well with the reliability challenges of large-scale distributed systems, composed of often unstable commodity hardware. The state daemon catches illegal or tardy state transitions. For example, if a node hangs while rebooting, the state daemon times out and attempts an alternate reboot mechanism.

An experiment is generated from a user's *ns* script or via a GUI, resulting in hard state in Netbed's relational database. It represents a network configuration, such as switch VLAN mappings or IP tunnels, and node configurations, including OS images. A node's local disk is currently considered soft state; this allows an experiment to be "swapped out" and later regenerated from the database onto other hardware. The database provides a single namespace and abstractions for heterogeneous resources. The resulting resource **transparency** enables simulated, emulated, live, or hybrid deployments to be similarly realized.

Netbed's **efficient** mapping of a virtual topology to physical resources enables an interactive style of exploration. With the aid of abstraction techniques, our simulated annealing algorithm overcomes the NP-complete local mapping problem within seconds. If users request wide-area paths with specific characteristics (such as latencies), a genetic algorithm finds paths that best match the data provided by periodic path monitoring.

Netbed's **virtualization** allows links to be treated independently of their physical realization. VLANs control the topology between emulated links, while interposed nodes emulate link properties. Netbed obtains wide-area nodes by managing MIT's "RON" nodes; isolation and control of wide-area links is optionally provided by the automated establishment of IP tunnels and routing. Simulated resources are integrated through *nse*, a version of *ns* that interacts with real traffic. At runtime, local and simulated resources can be controlled via a distributed event system; wide-area control is not yet implemented.

Node management services load disk images, boot, configure, and monitor nodes. These services include a highly tuned diskloader which exploits reliable multicast and domain-specific compression to load 3GB images onto hundreds of local nodes in 100 seconds. Wide-area node support provides secure, robust auto-configuration and image update. Such **automated** configuration replaces hours of manual labor with 5 minutes of hands-off automation.

An administrative subsystem supports a hierarchical authorization model that minimizes staff time and maps to natural organizational structure. Netbed's authentication subsystem automatically distributes `ssh` keys. Project-specific NFS or SFS trees are exported appropriately.

Conclusion: Netbed is uniquely valuable for studying distributed systems and networks. Additionally, the concepts and software apply to many other areas, including: *Reliability and fault-injection studies*: an ideal platform. *Storage systems*: Netbed provides configurable network-attached "smart disks." *Cluster management*: Netbed flexibly partitions a cluster, providing "virtual clusters" that traditional cluster software can then manage. This provides unique features such as complete isolation and choice of OS. *Analysis and debugging of distributed systems*: since Netbed can provide a closed distributed system, it can be extended to provide a low-level distributed virtual machine, allowing the system under study to be checkpointed, traced, and rolled back and forward in time. *Security*: leveraging its closure and isolation, Netbed could quarantine cyberattacks too dangerous to study in the real world.

Sponsors: NSF ANI-0082493, Cisco, DARPA F30602-99-1-0503. www.cs.utah.edu/flux/papers/netbedtr02.pdf, netbed@flux.cs.utah.edu.

Scalability and Accuracy in a Large-Scale Network Emulator

Ken Yocum, Kevin Walsh, Amin Vahdat, Priya Mahadevan,
Dejan Kostic, Jeff Chase, and David Becker *
Department of Computer Science, Duke University
{grant,walsh,vahdat,priya,dkostic,chase,becker}@cs.duke.edu

Experimental distributed services are difficult to evaluate due to their scale and the complexity of the Internet. The lack of an accepted evaluation methodology that predicts real behavior is a key obstacle to progress in distributed services research. ModelNet is a network emulation environment for evaluating distributed services across large-scale networks. A key research contribution of ModelNet is the development of a number of novel techniques for large-scale network emulation that trade accuracy for scalability.

ModelNet allows researchers to deploy unmodified software prototypes in a configurable Internet-like environment. Unlike live deployment, where researchers wait for extraordinary events to test a distributed application's resilience to harsh network conditions, ModelNet can reproduce those events, emulating flash crowds and network partitions. While simulation often abstracts away key details of real systems, emulation in ModelNet captures complex behavior by running unmodified application code and kernels at the end systems.

ModelNet emulates a wide-area network on a scalable gigabit LAN cluster. Edge nodes, running user-specified OS's and application software, are configured to route their packets through a set of ModelNet core nodes, which cooperate to subject the traffic to the bandwidth, latency, and loss profile of a target network topology. Target topologies are created from Internet traces (e.g., from CAIDA), BGP dumps, or synthetic topology generators (GT-ITM, Brite, or Inet).

Of course, no cluster can capture the full scale and complexity of the Internet. A significant contribution of this work is an evaluation of a set of scalable techniques for approximating Internet conditions. One technique, *distillation*, is applied to the target topology before running the emulation. Distillation applies a set of transforms to the target topology to decrease emulation cost. Another technique, *assignment*, distributes the emulation across the ModelNet cores. The goal of these techniques is to provide the highest emulation accuracy within the capacity of the available emulation hardware.

ModelNet distillation lowers emulation cost by reducing the number of hops packets take in the target topology. With respect to the measured overheads of our hardware, removing an average of two hops per packet can theoretically double the emulation capacity, in packets/sec, of a single core. There are two extremes to distillation. A hop-by-hop distillation is isomorphic to the target topology and provides the highest degree of emulation accuracy. Application cross traffic and contention on all links are emulated faithfully. In contrast, an end-to-end distillation creates a fully interconnected mesh between all clients. Accuracy suffers as cross traffic is isolated from previously affected flows. Distillation provides a continuum between these two extremes, trading accuracy for increasing scalability.

ModelNet assignment maps pieces of the distilled topology across the available number of ModelNet cores. The ideal assignment is dependent not only on the static information available in the distilled topology (routes, link rates and delays), but also upon the runtime load generated by the application. Assignment seeks to minimize the number of packets crossing between cores. This reduces both CPU and link utilization. Our current assignment techniques, based on graph partitioning algorithms, significantly reduce intercore traffic, decreasing the number of cores required for the emulation.

ModelNet cores are implemented as a set of loadable kernel modules for FreeBSD. Core nodes can accurately emulate 85,000 packets/sec at nearly 100% CPU utilization. At this load packets are delayed within a millisecond of link specifications. Experiments with several large-scale distributed services demonstrate the generality and effectiveness of the infrastructure: a peer-to-peer file service, an adaptive overlay, a replicated web service, and an ad hoc wireless communication scenario.

*This work is supported in part by the U.S. National Science Foundation (EIA-9972879), Hewlett Packard, IBM, Intel, and Microsoft. Vahdat is also supported by an NSF CAREER award (CCR-9984328)

DVSR: a High-performance Metro Ring Protocol

L. Balzano, V. Gambiroza, Y. Liu, P. Yuan, E. Knightly, S. Sheafor
ECE Department, Rice University, Vitesse Semiconductor
sunbeam, violeta, yhliu, pyuan, knightly@rice.edu, steve.sheafor@vitesse.com
http://www.ece.rice.edu/networks/DVSR

The Resilient Packet Ring (RPR) IEEE 802.17 standard is under development as a new high-speed backbone technology for metropolitan area networks. A key performance objective of RPR is to simultaneously achieve high utilization, spatial reuse, and fairness, an objective not achieved by current technologies such as SONET and Gigabit Ethernet or by legacy ring technologies such as FDDI. The core technical challenge for RPR is the design of a bandwidth allocation algorithm that dynamically achieves these properties. The difficulty is in the distributed nature of the problem, that upstream ring nodes must inject traffic at a rate according to congestion and fairness criteria downstream. Unfortunately, the proposed algorithms in the current draft standards have a number of critical limitations. For example, we show that in a two-flow two-link scenario with unbalanced and constant-rate traffic demand, a draft RPR algorithm will suffer from dramatic bandwidth oscillations within nearly the entire range of the link capacity. Moreover, such oscillations hinder spatial reuse and decrease throughput significantly. We are developing a new dynamic bandwidth allocation algorithm called Distributed Virtual-time Scheduling in Rings (DVSR). The key idea is for nodes to compute a simple lower bound of temporally and spatially aggregated virtual time using per-ingress counters of packet (byte) arrivals. We show that with this information propagated along the ring, each node can remotely approximate the ideal fair rate for its own traffic at each downstream link. Hence, DVSR flows rapidly converge to their ring-wide fair rates while maximizing spatial reuse. To evaluate DVSR, we are using a combination of theoretical analysis, simulation, and implementation, including a proof-of-concept prototype of DVSR on a 1 Gb/sec network processor testbed.

Modeling of Mobility-Induced Losses in MANETs (MILMAN) *

Karthik Dantu Shyam Kapadia Rishi Sinha † Ahmed Helmy ‡

Introduction : We study the effects of mobility on packet loss in mobile ad-hoc networks (MANETs). Mobility is generally accepted as a cause of packet loss. Other broad causes of loss in MANETs are losses in the wireless channel and losses due to congestion. While there exist many models for wireless channel and congestion losses, mobility-related losses have not been studied in depth. Past studies have investigated the effect of mobility on actual transport-layer throughput, an approach that does not capture the character of losses that occur and are then corrected by the transport layer. Our approach is to measure the likelihood of a packet being affected by mobility-related loss, rather than to measure the throughput achieved by any particular MANET transport layer.

Loss Metric : A mobility-related packet loss may occur in one of many different ways. In order to distinguish mobility-related packet loss from other kinds of packet loss, we need to decide which mechanisms at which layers of the protocol stack are responsible for this kind of loss. This involves assuming particular link layer and routing protocols. In our simulations, we use 802.11 and DSR respectively. A challenge we face is that some kinds of mobility-related losses occur through similar mechanisms as losses due to congestion (namely, buffer overflow and timeout). We have identified four mechanisms that cause the complete set of losses that we attribute to mobility. [1] Drops at intermediate hops because the source route was invalid and salvaging did not succeed. [2] Drops due to send failure at the source. [3] Drops from the source's send buffer because of timeout. [4] Drops at the source's send buffer due to insufficient space to queue all packets which are pending route discovery. We include losses due to these four causes in the loss metric we define, the *MILMAN* metric. In addition, we count the number of packets that had their original source routes modified en-route in the salvaging process, and reached the destination successfully. Though these packets are not lost, they are clearly affected by mobility, and we include them in the metric. Through experimentation, we found that this component contributes a very small percentage to the metric.

Connectivity Metric: We studied the correlation of the MILMAN metric with a metric that measures the number of changes in the shortest path between the source and destination on the connectivity graph. We find that there is weak positive correlation for dense graphs, but strong negative correlation for sparsely connected graphs. (Figure 1). However, we note that for sparse networks, the above metric has negative correlation with velocity, and hence may not be representative of *mobility*. We considered another metric, the disconnection time between the source and destination and as expected we found that the MILMAN metric varies directly with this metric(Figure2).

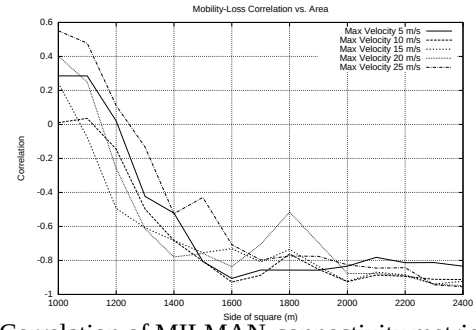


Fig 1. Correlation of MILMAN-connectivity metric vs.area

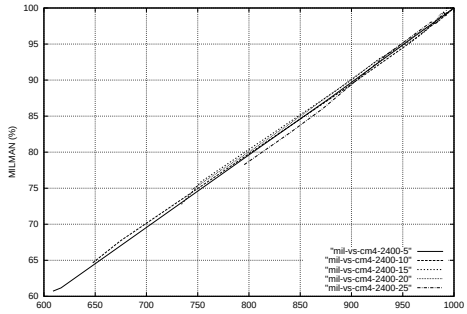


Fig 2. MILMAN vs.Disconnection time

Setup : Simulations were conducted in ns-2 with the random waypoint mobility model for CBR traffic. A single source-destination pair was used for each of the simulations. The area was varied from 1000sq.m. to 2400 sq.m. at intervals of 100 sq.m. and maximum velocity of nodes from 5 m/s to 25 m/s in steps of 5. Ten different scenarios were chosen for each area-velocity combination. Hence, a total of 750 simulations were conducted.

Results and Future Work : Our preliminary results indicate that though an increase in mobility(velocity) leads to greater packet loss in densely connected networks, we can observe some interesting (though isolated) cases where it leads to a *decrease* in packet loss in case of sparsely connected networks.

We plan to study the characteristics of these interesting cases, losses in other mobility models and routing protocols as well as formulating models for mobility-related losses for particular mobility models, with the goal of being able to use the model in the performance analysis of specific transport layers.

*MILMAN website. *http://nile.usc.edu/ee499/milman*

†Karthik, Shyam and Rishi are affiliated to Computer Science Dept. University of Southern California, LA - 90089. They can be reached at {dantu|kapadia|rishisin}@.usc.edu

‡A. Helmy was supported by NSF CAREER Award 0134650.He is affiliated to Electrical Eng. Department, University of Southern California, LA - 90089. He can be reached at helmy@ceng.usc.edu

Fixing 802.11 Access Point Selection*

Glenn Judd and Peter Steenkiste
Carnegie Mellon University
{glennj, prs}@cs.cmu.edu
http://www.cs.cmu.edu/~glennj/scp/FixingAPSelection.html

As 802.11 deployment and use become ubiquitous, it becomes increasingly important to make efficient use of the bandwidth provided. Unfortunately, 802.11's current access point selection model fails to enable efficient use of access point bandwidth. We discuss why the current 802.11 access point selection model fails, and evaluate alternative models via simulation. Our results indicate that with a small amount of enhanced functionality, 802.11 access point selection can be vastly improved.

As motivation consider the following real-life example: two large lecture halls in Carnegie Mellon University's business school were covered by a single access point that was regularly overloaded. In order to remedy this situation, an additional access point was added, on an available channel, to cover the same area. The result: despite measures to ensure that access point coverage was complete and balanced, the original access point remained overloaded while the new access point picked up a minimal load.

What went wrong? The answer lies in 802.11's minimal support for access point selection. The standard currently merely provides a mechanism for association and disassociation. No information regarding access point load is provided; as a result, the only viable policy for access point selection is to pick the access point with the best signal-noise ratio. An analysis of the signal-noise ratios in the two lecture halls reveals that while the access point coverage is nearly equal, for most locations the original access point has a slightly better signal-noise ratio. The result is that nearly all mobile hosts associate with the original access point leaving it overloaded and the new access point nearly idle.

*This research was sponsored in part by the Defense Advanced Research Project Agency and monitored by AFRL/IFGA, Rome NY 13441-4505, under contract F30602-99-1-0518. Additional support was provided by Intel. Glenn Judd is supported by a DoD National Defense Science and Engineering Graduate (NDSEG) Fellowship.

While others have publicly proposed solutions to this problem, their solutions have suffered from significant shortcomings such as the need for global load and user information, the failure to consider signal quality, and the need for substantial functionality that is not present in 802.11. We introduce a load sensitive approach to access point selection that is distributed in nature, and amenable to implementation in 802.11 networks with minimal modification to the standard.

Unfortunately, evaluation of this approach is non-trivial. 802.11 access point selection is implemented in firmware which is difficult to modify, and evaluation on a large scale in an operational network is impractical. To overcome these obstacles, we have developed a simulator that utilizes extensive samples of access point signal quality, a model of the physical world, traces of network activity, and traces of user access-point association (used to develop "synthetic models" of user movement through physical space). Using this simulator, we can recreate (in an approximate fashion) the operation of the wireless network during a given period of time. This framework allows us to compare the effectiveness of various access point selection models.

Our preliminary experiments have been aimed at evaluating access point selection mechanisms in the context of the lecture hall scenario described previously. Using network traces from 4pm to 5pm hour on April 30th, 2002 we evaluated the current "SNR Only" access point selection algorithm versus a "Load Sensitive" access point selection algorithm. Under our algorithm, mobile hosts select an access point based on both the current signal-noise ratio as well as the current load at the access point. To avoid oscillation we introduce randomness and hysteresis. Our results show that, as expected, taking load into consideration allows balancing of access point utilization, and provides users with increased network performance.

Efficient Resource Discovery for Large Scale Ad hoc Networks using Contacts

Nitin Nahata, Priyatham Pamu, Saurabh Garg, Ahmed Helmy*
Department of Electrical Engineering, University of Southern California
{nnahata, pamu, sgarg, helmy}@usc.edu - http://nile.usc.edu/ee499/contacts

The infrastructure-less nature of ad hoc networks renders resource discovery a challenging problem. Unlike wired networks, mobility induces frequent route changes. Traditional protocols proposed for resource discovery in such environments either involve global flooding or are based on complex hierarchy formation. While flooding does not scale well, hierarchy formation involves complex coordination between nodes and may suffer serious performance degradation due to mobility.

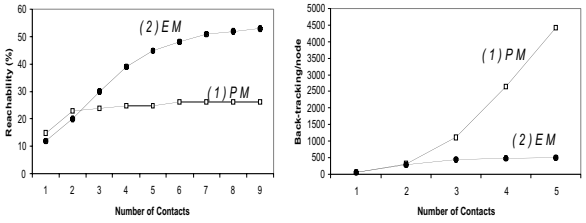
To overcome these limitations we propose a new architecture based on the concept of *small worlds*. In our architecture we adopt a hybrid approach in which a node uses *(a)* proactive routing (e.g., DSDV) to reach its ***neighborhood*** within a limited number of hops, *R*, (typically 3-5 hops), and *(b)* reactive querying beyond the neighborhood via ***contacts***. *Contacts* act as *short cuts* that attempt to transform the network into a small world by reducing the degrees of separation. They help in providing a view of the network beyond the neighborhood during resource discovery. Each node maintains state for a few contacts (typically 5-15) beyond its neighborhood. Contacts are polled periodically to validate their presence and routes. For discovering resources efficiently, queries are sent to the contacts that leverage the knowledge of their neighborhood. Our approach has the following design goals: scalability to thousands of nodes, efficiency in terms of network overhead, robustness and decentralized operation. We do not assume availability of any geographic information.

Contact selection (*CS*) is initiated by a source node, *S,* by sending contact-selection queries (*CSQ*) through nodes at the edge of its neighborhood (edge nodes). *CSQ* contains source ID, hop count and list of *S* s previous contacts. Our first *CS* protocol, the *P method (PM),* states that a node receiving the query, checks whether any of the *S* s previous contacts lie inside its neighborhood. This reduces overlap between contact neighborhoods and hence improves reachability. If no contact lies in its neighborhood, the node chooses to be a contact with probability *P*, otherwise it forwards the query to a neighbor. $P=(H - 2R)/(r_{max} - 2R)*100$ where *H* is hop count from *S* and r_{max} is the maximum distance (in hops) at which a contact can be selected. The above equation helps to locate contacts that lie between *2R* and r_{max} hops from the source and reduces overlap between neighborhoods of *S* and the contact. If r_{max} is reached and a contact is not chosen (e.g., due to overlap or use of *P*), a *back-tracking* mechanism is used to select another contact. Our study shows that overhead of such mechanism may be significant. To ameliorate such a problem we propose our second *CS* protocol, the edge method, *EM*. In *EM* the list of edge nodes (*E-List)* for *S* is also included in the *CSQ*, but *P* is not used. This reduces back-tracking traffic drastically. A node receiving *CSQ* checks if any node on the *E-list* lies inside its neighborhood. This ensures non-overlap with *S* s neighborhood. Also, in *EM,* query and node IDs are kept to prevent loops. *EM* resulted in notable increase in reachability with drastic decrease in overhead over *PM,* as shown in Fig. 1.

Contact maintenance is based on *S* periodically polling its contacts to validate and update their routes. Broken links are recovered using neighborhood knowledge (***local recovery***). Local recovery reduces maintenance overhead and betters protocol robustness.

When a source node, *S,* initiates a resource discovery for a target, the target is looked up in *S* s neighborhood, then a query is sent to *S* s contacts, which in turn look in their neighborhood. The contacts may further query their contacts, so on, up to the ***query depth***. This creates *S*-rooted tree of contacts, which is helpful in making our approach scalable.

We carried out simulations on various networks (100-1000 nodes) under mobility, and analyzed performance of our architecture in terms of reachability and contact selection and maintenance overhead. We found that as the number of contacts increased, both reachability *and* overhead increased. This shows a clear tradeoff between reachability and overhead (shown in Fig. 1). Also, increasing *R* increased reachability but led to more proactive overhead inside the neighborhood. Increasing query depth improved reachability and seems scalable. We also noticed that maintenance overhead decreased over time as the nodes seemed to find more relatively stable contacts. We plan to study effects of various mobility models on maintenance overhead.



(a) Reachability (b) Back-tracking Overhead

Fig. 1. Reachability vs. Overhead trade-off for (1) *P* method, *PM,* protocol, and (2) Edge method, *EM,* protocol. (shown for 500 nodes, 710mx710m, transmission range=50m, *R=3, r_{max}=20, query depth=1*).

* A. Helmy was supported by NSF CAREER Award 0134650.