the nature of the beast: recent traffic measurements from an Internet backbone

k claffy
caida
kc@caida.org

Greg Miller and Kevin Thompson
MCI/vBNS
gmiller,kthomp@mci.net

Contents

---

abstract

As described in last years Inet ’97 paper [1], MCI has implemented a high-performance, low-cost monitoring system that can monitor Internet traffic (cell/packet headers) and

perform analyses, and deployed them on OC-3 trunks within iMCI's backbone and also within the NSF-sponsored vBNS (very High performance Backbone Service). This publicly-available tool facilitates measurement and analysis of high-speed OC-3, and now OC-12, trunks that carry hundreds of thousands of simultaneous flows. As a follow up to last year's paper, we provide some new data analyses as well as comparisons with last year's data that may suggest trends in changing workload profiles. All the data in this paper is based on recent wide-area MCI Internet backbone traffic as recorded by the Coral monitors.

---

the nature of the beast: recent traffic measurements from an Internet backbone

Everything you've learned in school as "obvious" becomes less and less obvious as you begin to study the universe.
For example, there are no solids in the universe.
There's not even a suggestion of a solid. There are no absolute continuums.
There are no surfaces. There are no straight lines.
---- *R. Buckminster Fuller*

Introduction

Sustained, rapid growth, increased economic competition, and proliferation of new applications have combined to change the character of the Internet in recent years. The sheer volume of the traffic and the high capacity of the trunks have rendered traffic monitoring and analysis a more challenging endeavor. As described in last years Inet '97 paper [1], MCI has implemented a high-performance, low-cost monitoring system that can monitor Internet traffic (cell/packet headers) and perform analyses, and deployed them on OC-3 trunks within iMCI's backbone and also within the NSF-sponsored vBNS. This publicly-available tool facilitates measurement and analysis of high-speed OC-3, and now OC-12, trunks that carry hundreds of thousands of simultaneous flows. As a follow up to last year's paper, we provide some new data analyses as well as comparisons with last year's data that may suggest trends in changing workload profiles. All the data in this paper is based on recent wide-area MCI Internet backbone traffic as recorded by the Coral monitors.

We reveal the characteristics of the traffic in terms of flow size by protocol, percentage composition of traffic by protocol and application, distributions of flow sizes, length of packet trains, and statistics on IP fragmentation, prefix length distribution, and address space utilization. Where applicable, we also

compare to the data from August 1997 as described in the previous study [2].

Coral measurement architecture

The original Coral DOS architecture was described in detail in Apisdorf, *et al.*'s Inet '97 paper [1]. The goal of the Coral project is to address three incompatible trends: (1) Current, widely-used statistics gathering tools, which are largely FDDI and Ethernet based, have difficulty scaling to higher speeds; (2) ATM circuits at OC-3 and higher rates are increasingly used for high-volume backbone trunks and interconnects; and finally (3) detailed, flow-based analysis is important to understanding usage patterns and growth trends, but such analysis is not generally possible with the data that can be obtained directly from today's routers and switches.

The current Coral implementation satisfies the need for a high-speed monitoring and flow analysis tool while meeting the project's two driving design constraints of flexibility and low cost. Coral is a programmable data collection and analysis tool that can be easily modified as we codify and refine our understanding of the desired statistics. Furthermore, it is inexpensive to build, which facilitates widespread deployment. Both the flow analysis code and monitor architecture are in the public domain. Work is underway to expand the family of Coral monitors to address different speeds (OC12, OC48, DS3), interface types (e.g., Digital gigaswitch), operating systems (e.g., UNIX).

Coral hardware

The Coral/OC3mon platform is an IBM personal computer clone with 256 MB of main memory, a 166 MHz Intel Pentium processor, an Ethernet interface, two ATM interface cards, and a 33 MHz 32-bit-wide PCI bus. The ATM interface card used in the current Coral/OC3mon implementation is the Fore Systems ATM network interface card (NIC) for the PCI bus. The Intel i960 processor on this interface card allows us to optimize Coral/OC3mon operation with custom firmware.

We attach the two Coral/OC3mon ATM NICs to an OC-3 optical fiber pair carrying IP-over-ATM traffic. We connect the receive port of each ATM card to the monitor port of an optical splitter. The splitter carries a fraction of the light from each fiber to the receive port of one NIC. Attached to an OC-3 trunk that terminates on a switching device (e.g., an ATM switch or a router), one of the Coral/OC3mon NICs sees all traffic received by the switching device and the other NIC sees all traffic transmitted by the switching device. The Coral/OC3mon NICs capture traffic on the two directions of an OC-3 link independently.

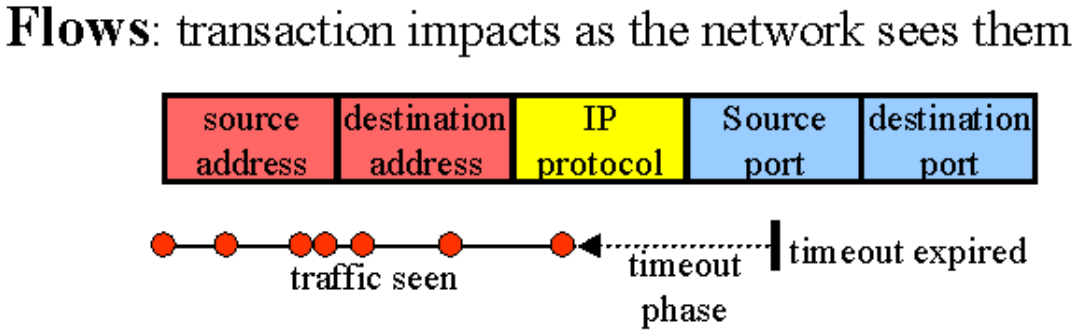
Coral software

The custom-developed Coral/OC3mon firmware is implemented in C++ and assembly code and provides full flexibility in terms of collection and analysis capability. Coral/OC3mon supports three modes of data collection: raw cell trace capture, active flow reporting, and expired flow analysis. In raw trace mode, Coral/OC3mon captures either every cell, or the first cell of every packet (AAL5 frame), that appears on the link. In this mode, Coral/OC3mon does not analyze captured data, but simply produces a time-stamped raw cell trace. The maximum length of a raw cell trace collected by OC3mon is limited to the amount of RAM in the monitor. In the two flow analysis modes, OC3mon collects statistics regarding flows, either active or expired, where the definition of a flow is configurable. Note that Coral processes every cell/packet, there is no statistical sampling involved in the monitoring itself.

Measurement methodology

We define a flow as a uni-directional traffic stream with a unique [source-IP-address, source-port, destination-IP-address, destination-port, IP-protocol] tuple (See figure 1). Any flow for which the monitor has not seen a packet within the last 64 seconds is considered to be an expired flow. A flow for which a packet was seen within the last second is considered to be an active flow. The monitor reports statistics based on expired flows whenever polled by a collector. After reporting the expired flows statistics to the collector, the monitor clears its stored state on all expired flows. It continues to maintain state information on flows that have not yet expired. These continuing flows, which have not yet been reported on, are referred to as known flows. The data presented in this paper is derived from expired flows statistics as reported by the monitors when being polled around the clock on 5-minute intervals. Flows that have been in progress (known) for one hour are artificially expired by the monitor so they can be reported. This mechanism for delimiting long-lived flows can affect the data that is reported by producing artificial traffic spikes.

figure 1: flow framework *courtesy Hans-Werner Braun, NLANR/MOAT*



The monitor does not maintain statistics on expired flows individually, but instead aggregates them on a per-protocol basis. For this reason, flow statistics such as duration, byte volume, and length in packets are reported in terms of averages. Future plans include expanding the monitor's functionality to support collection of more detailed distribution information, as we recognize that averages can be of limited value in describing distributions of Internet traffic characteristics because of their wide variation.

FreeBSD Unix port

In response to community feedback, NLANR's (National Laboratory for Applied Network Research) Measurement and Operations Analysis Team (MOAT) [5] and the Cooperative Association for Internet Data Analysis (CAIDA) [4] have ported Coral/OC3mon to FreeBSD Unix (2.2.2-RELEASE). The Unix port is divided into two portions, the kernel-space device driver which deals with the Fore PCA-200E card directly and the user-space code that does the flows analysis and fulfills requests for summary reports. The Unix version differs slightly from the DOS version in the three different modes of data collection; for details see the Coral home page [3]. Briefly, in the packet trace collection mode, Coral/DOS requires a manual process to start, terminate, and copy data from the monitor to a host. The Unix version is more amenable to automation since cron utilities are available from the operating system.

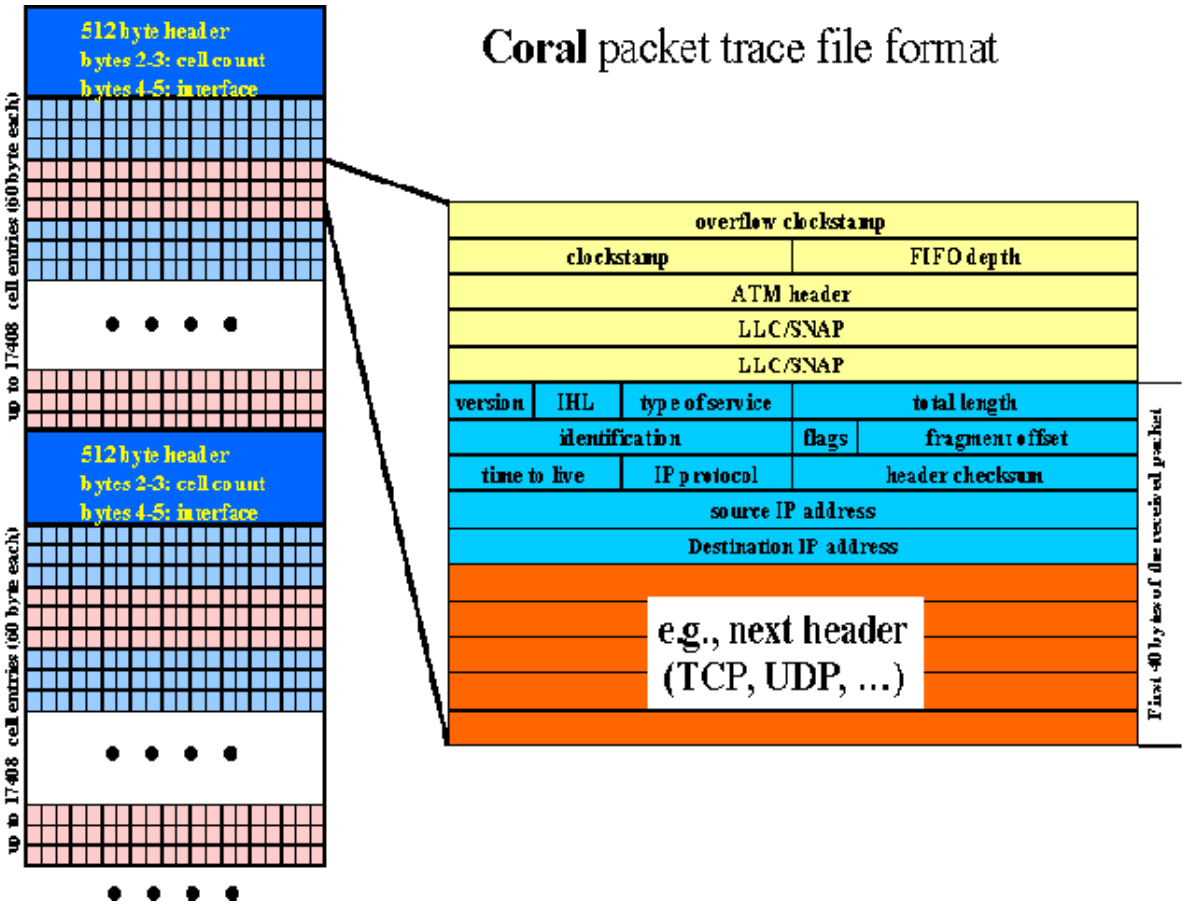
In flow analysis summary mode, Coral/DOS results are available by telnet to a port on the box; Coral/Unix requires that one manually send the process a signal.

Management and security issues also differ on the two platforms. There is as yet no encryption of any communication with the Coral/DOS machine (access, management, or data transfers); management is via out-of-band (terminal-based) access. These characteristics may limit Coral/DOS deployment in a broad environment with needs for remote management. In contrast, Coral/Unix uses a standard LAN based connection to the Internet. While more conducive to many environments that might want tools for scheduling, encryption, and basic operating system support for a monitoring platform, the Coral/Unix implementation is less stable than the DOS version and is still undergoing testing.

Packet header trace format

The Coral output format for a raw trace of packet headers consists of approximately 1MB concatenated blocks of data. Each data block consists of a 512-byte block header, followed by 17,408 packet entries of 60 bytes each. The block header includes information about which Coral Monitor interface the data in this block is from, and how many of the 17,408 entries are actually used. The 60-byte entries are depicted below. The DOS and Unix/Coral versions differ in that the DOS version has the first 64 bits of each 60-byte entry byte-swapped. A flag to indicate byte-swapping resides in the first 16 bits of each block header, with -1 (0xffff) denoting the non-byte-swapped (Unix) version.

figure 2: Coral packet trace format (byte-swapped version) *courtesy Hans-Werner Braun, NLANR/MOAT*



CAIDA/NLANR also provides scripts to convert from Coral format to ASCII, from Coral format to

libpcap/tcpdump format [10], and to privatize source/destination IP addresses, all available from CAIDA's web site [3].

OC12mon

As MCI's backbone transitioned from OC-3 to OC-12 speeds, they needed a new version of the Coral monitor. The new ATM OC12 cards, designed by Applied Telecom, satisfy constraints of performance and public code availability. The daughter card on the Applied Telecom PCI card for OC12MON can be replaced with one that does OC-3, including packet-over-SONET, if consistency of monitors across link speeds or the features of the Applied Telecom firmware are desirable.

OC12mons, now undergoing field tests on iMCI links, are designed to collect the first 2-3 cells per packet, (3 cells yields 144 bytes of payload including LLC/SNAP or other headers), which is useful for tcpdump/snoop type functionality. Writing to disk while monitoring at line speed, an important feature of the OC12mons to permit the capture of arbitrarily long raw traces, requires disk write speeds of 80MB/s throughput. As a result, under current technology, an OC12mon implementation requires a highly tuned RAID array using FibreChannel or equivalent capabilities.

All other hardware specifications for OC12mon are the same as for OC3mon, including optical splitter, monitor, keyboard, ethernet interface, etc. MCI uses Pentium Pro CPUs for OC12mon, with no perceived problems. (Note that PCI bus is preferable to ISA here, since the latter tend to hold onto the bus longer.) Specifications and code are available from the Coral home page [3].

Measurement points

Two OC3mons are installed on OC-3 links within nodes on the iMCI backbone. Each OC3mon monitors traffic on a fiber pair between a core router and a backbone ATM switch. The first point serves as a junction for several backbone trunks as well as an access point for local customer traffic near a major U.S. East Coast city. The second point includes a U.S.-U.K. trans-Atlantic DS-3 trunk, where the monitor sits on an OC-3 fiber pair between the router to which the international DS-3 trunk is homed and the backbone ATM switch. For both points our measurements are taken on 13 April 1998, from midnight to midnight. Some additional raw trace data was taken from the domestic trunk for in depth analysis not possible from the flow data alone.

Results

Tables 1 and 2 provide summary statistics on the highest volume TCP and UDP applications for the 24-hour period of Monday 13 April 1998. Table 1 shows the top ten TCP applications from the international measurement point sorted by byte volume. **There were 2.3 billion total packets and 0.8 trillion bytes for the day. The top ten TCP applications constituted 122 million flows, 1.867 billion packets and 708.440 billion bytes. UDP traffic constituted 17 million flows, 0.1 billion packets, and 14 billion bytes.**

Table 1: top ten TCP applications sorted by byte volume
(24 hours of Monday 13 April 1998)

Proto	src port	dest port	flows	packets	bytes
TCP	http	0	48766857	682980223	5.13e+11
TCP	0	http	69961315	935240039	7.38e+10
TCP	0	nntp	78521	57621854	3.82e+10
TCP	ftp-data	0	123702	31144523	2.97e+10
TCP	0	smtp	1632969	45616152	2.33e+10
TCP	nntp	0	70606	39495541	1.40e+10
TCP	5501	0	5319	5894572	7.27e+09
TCP	0	ftp-data	180794	29531305	3.68e+09
TCP	7070	0	27453	4799848	3.48e+09
TCP	smtp	0	1560102	34801980	2.01e+09

TCP TOTALS 2.054457e+09 packets and 7.775899e+11 bytes
TOTALS 2.337206e+09 packets and 8.144167e+11 bytes

Table 2: top ten UDP applications sorted by byte volume
(24 hours of Monday 13 April 1998)

Proto	src port	dest port	flows	packets	bytes
UDP	domain	domain	13830555	44310822	4.72e+09
UDP	7648	7648	6270	5099081	1.61e+09
UDP	27910	0	416481	8896619	1.60e+09
UDP	0	27910	613932	19672069	1.21e+09
UDP	dtspc	dtspc	195932	20327183	1.17e+09
UDP	27500	0	283211	7962525	9.40e+08
UDP	22555	22555	15655	7520126	9.17e+08
UDP	domain	0	1648306	3476921	7.25e+08
UDP	0	27500	325784	11800234	6.64e+08
UDP	0	ntp	98739	8581132	6.52e+08

UDP TOTALS 2.497650e+08 packets and 2.938063e+10 bytes
TOTALS 2.337206e+09 packets and 8.144167e+11 bytes

Packet sizes

figure 3 : **(a) distribution of packet sizes**

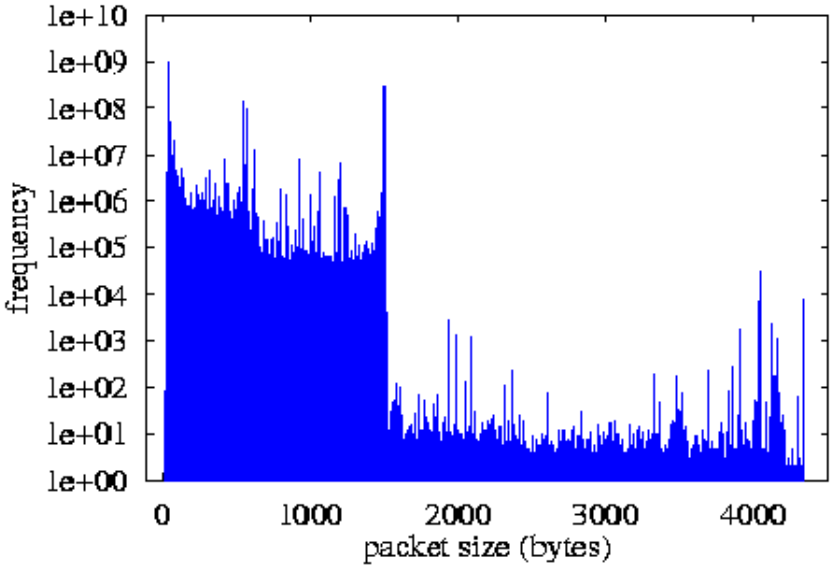


figure 3: (b) cumulative distribution of packet sizes, and of bytes by the size of packets carrying them

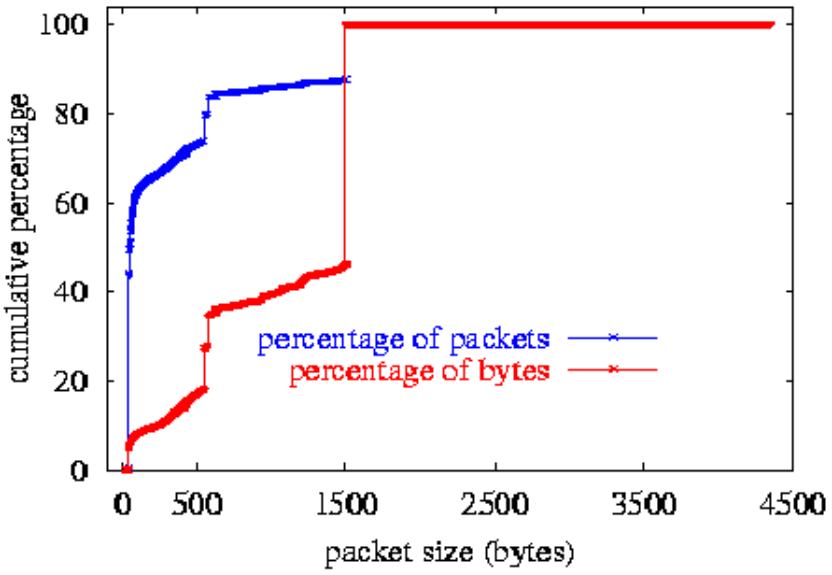


Figure 3a shows the distribution of packet sizes from a 24-hour time period on both directions of the measured trunk. As with graph from previous years [1], this figure illustrates the predominance of small packets, with peaks at the common sizes of 44, 552, 576, and 1500 bytes. The small packets, 40-44 bytes in length, include TCP acknowledgement segments, TCP control segments such as SYN, FIN, and RST packets, and telnet packets carrying single characters (keystrokes of a telnet session). Many TCP implementations that do not implement Path MTU Discovery use either 512 or 536 bytes as the default Maximum Segment Size (MSS) for nonlocal IP destinations, yielding a 552-byte or 576-byte packet size [11]. A Maximum Transmission Unit (MTU) size of 1500 bytes is characteristic of Ethernet-attached hosts.

Figure 3b shows the cumulative distribution of packet sizes, and of bytes by the size of packets carrying

them. This graph shows that almost 75% of the packets are smaller than the typical TCP MSS of 552 bytes. Nearly half of the packets are 40 to 44 bytes in length. Note however that in terms of bytes, the picture is much different. While **almost 60% of packets are 44 bytes or less, constituting a total of 7% of the byte volume**, **over half of the bytes are carried in packets of size 1500 bytes or larger**.

Composition of traffic by protocol

figure 4: figure composition of byte volume by IP protocol (left) and TCP application (right). *Data from 24-hour period on 13 April 1998.*

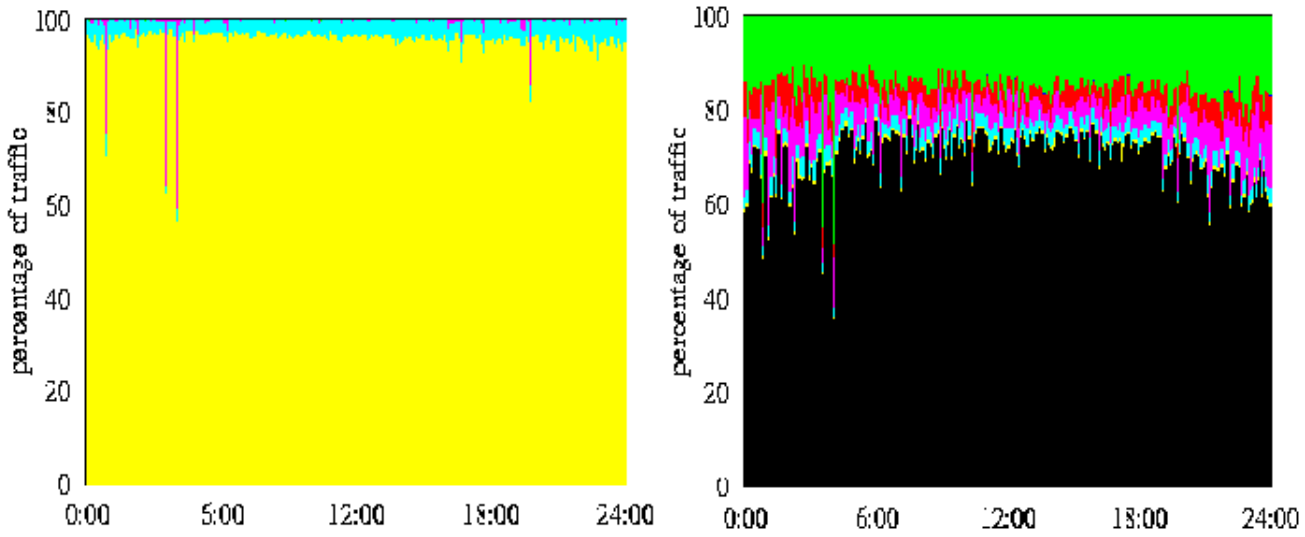


figure 5: composition of packets by IP protocol (left) and TCP application (right) *Data from 24-hour period on 13 April 1998.*

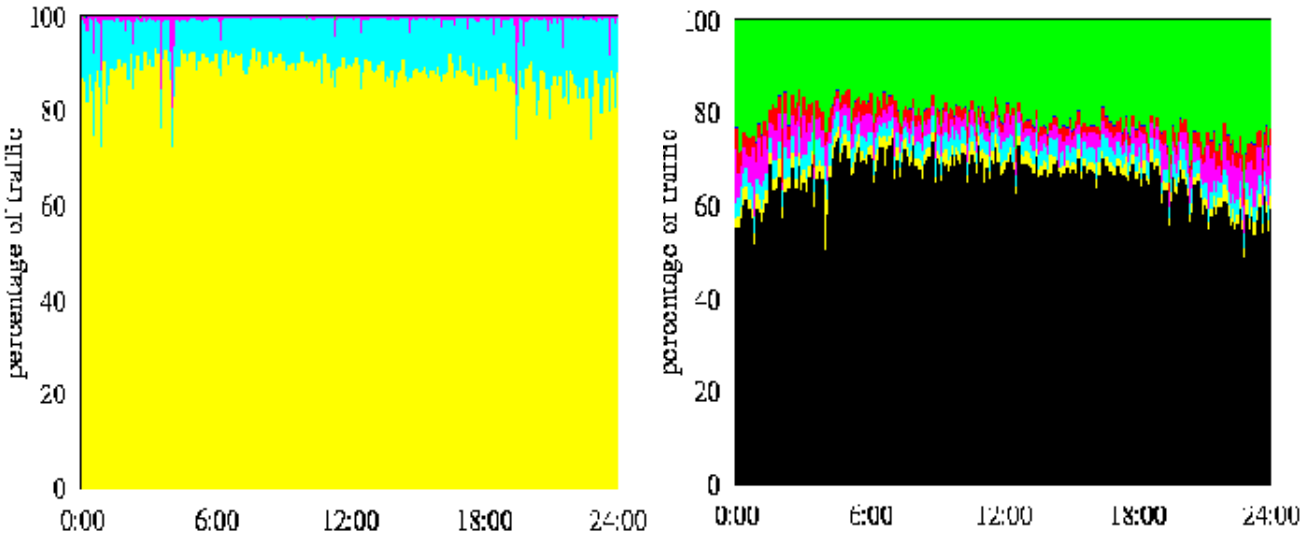
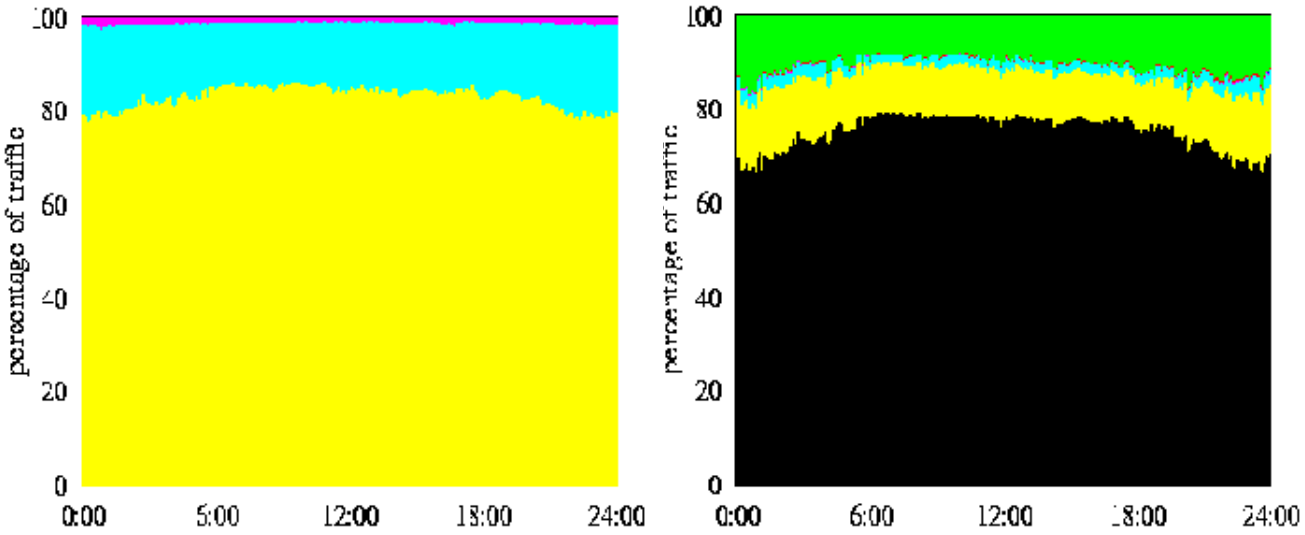


figure 6: composition of flows by IP protocol (left) and TCP application (right) *Data from 24-hour period on 13 April 1998.*



The graphs in figures 4-6 indicate the composition of the traffic over a 24-hour period on the measured link. Aside from a slightly greater proportion of web traffic, this data is not significantly different from measurements taken August 1997 [2]. The left-hand plots show traffic in terms of IP protocols; on the right are breakdowns by TCP and UDP applications. Focusing first on IP protocols, we observe that **TCP still by far dominates the traffic mix**. Over the course of a day, **TCP averages about 95% of the bytes, 90% of the packets, and 80% of the flows on the link**. UDP is the second largest category, at roughly 5% of the bytes, 10% of the packets, and 18% of the flows on average. The other IP protocols plotted are tunneled IPv6, encapsulated IP (IP-in-IP), ICMP, and an aggregate category for the remaining protocols labeled *'other'*. These other protocols individually make up a negligible percentage of the overall traffic. **ICMP** constitutes the third highest packet percentage after TCP and UDP, but still **makes up less than 2% of the overall packets and 0.5% of the overall bytes**.

The figures on the right side depict the proportion of the most prevalent TCP and UDP applications measured over a 24-hour period. For each application, we combine client-to-server and server-to-client (and in the case of DNS, server-to-server) traffic into a single category. We see that the **Web is the dominant application on the link, comprising up to 75% of the bytes, 70% of the packets, and 75% of the flows when client and server traffic are considered together**. In measuring applications, we end up with a larger *'other'* category than when measuring IP protocols. The *'other'* category is spread among a wide range of TCP and UDP port numbers, no one of which represents a significant percentage of the traffic by itself. Among the most common port numbers in this category are 81, 443, 3128, 8000, and 8080, which are all Web-related, indicating that the **Web may actually be slightly under-represented in our measurements**.

In addition to Web traffic, we identify five other applications that contribute an appreciable percentage of traffic: DNS, SMTP, FTP (data connections), NNTP, and Telnet. In terms of flows, DNS traffic represents the second largest application at nearly 18% of the overall flows. However, DNS flows are small, accounting for less than 3% of the total packets and 1% of the bytes on average. SMTP averages 5% of the bytes, 5% of the packets, and 2% of the flows. FTP data connections, on average, constitute roughly 5% of the bytes, up to 3% of the packets, and less than 1% of the flows. NNTP represents 2% of the bytes, and less than 1% of the packets and flows. Finally, **Telnet accounts for about 1% of the packets, and less than 1% of the bytes and flows, a marked decrease from recent years as alternative interactive protocols (e.g., ssh, kerberos, rlogin) have increased in popularity**[8,9].

Per flow averages (overall and per application)

figure 7: distribution of packets per flow

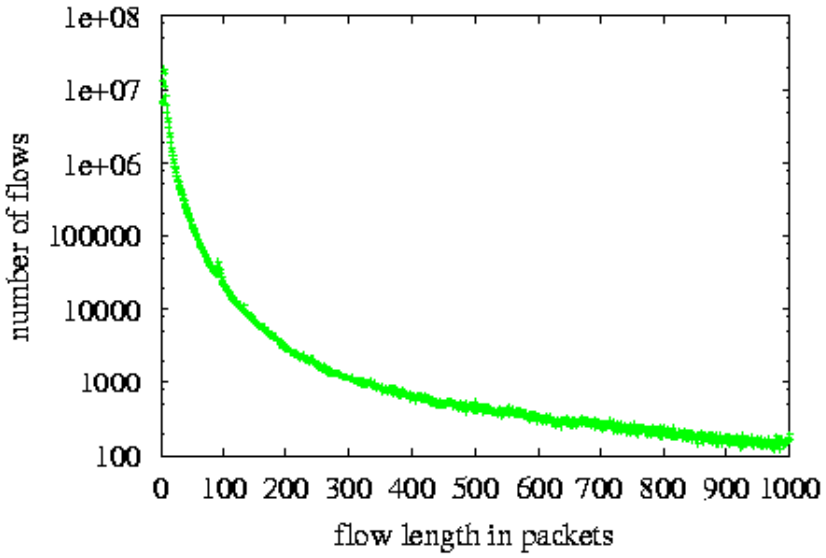
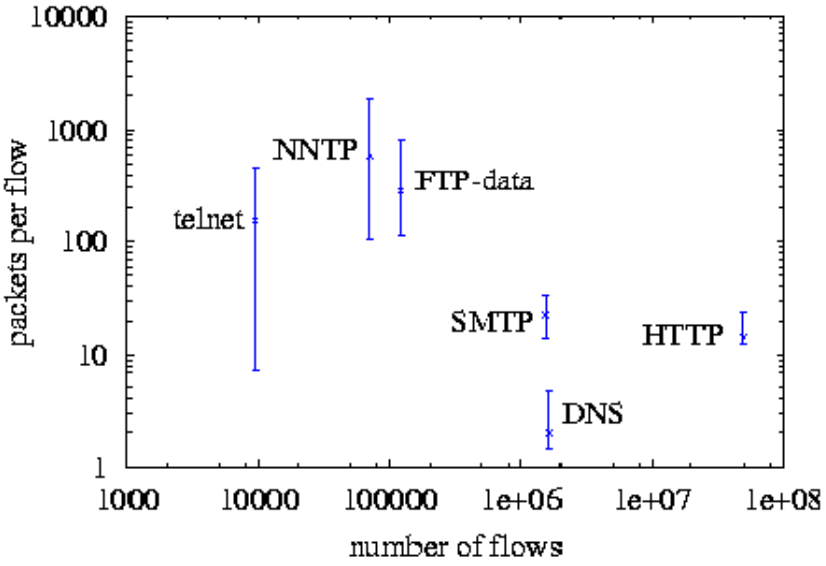


figure 8: distribution of packets per flow by protocol (log-log scale)



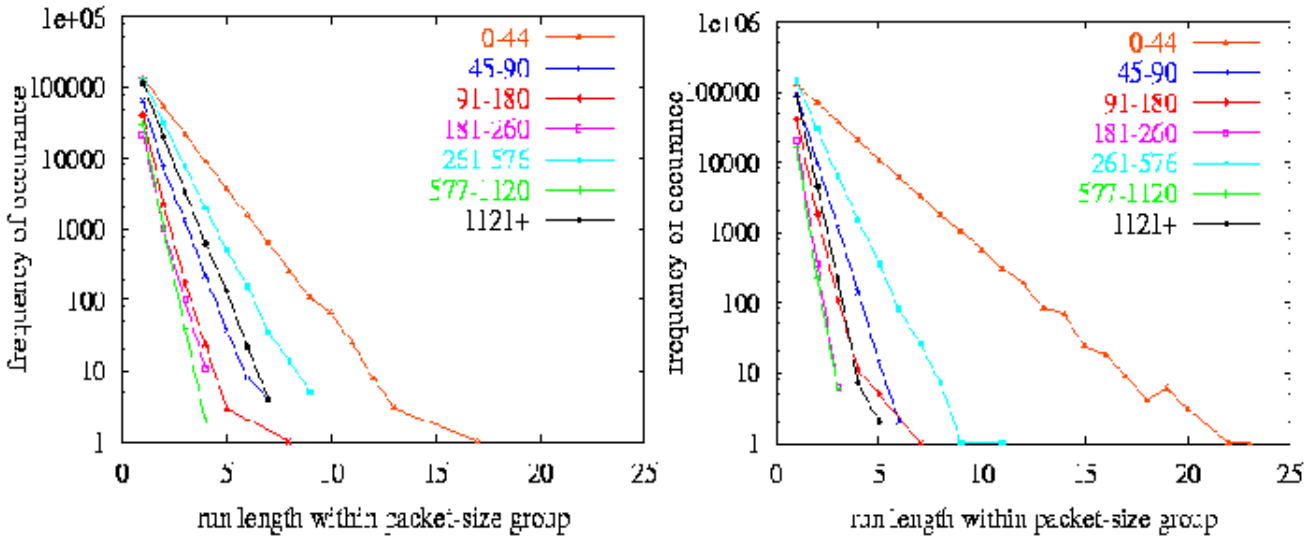
Figures 7 and 8 show the sizes of flows as measured in packets. Figure 7 graphs the packet-per-flow distribution for all flows; the distribution is remarkably long tailed; we truncate it here for viewability. Figure 8 focuses on the number of packets per flow as a function of specific TCP or UDP application. The vertical lines are box and whisker plots: the x represents the mean number of packets per flow for a 24-hour period. The top and bottom of the vertical lines indicate the maximum and minimum 5-minute averages over the 24-hour period, respectively. This figure shows on a log-log scale how small most of the transaction-style flows, e.g., HTTP, SMTP, DNS, are in contrast to the bulk data transfer-style flows, e.g., FTP-data, NNTP. Note the telnet flows can be composed of large number of packets but are much smaller in byte payload (graph not shown here).

Lengths of packet troops

We illustrate the behavior of sequences of packets categorized by packet sizes. Since looking at all packet sizes inhibits useful visualization and does not add significant benefit over bucketing packet sizes (dividing the range of packet size up into contiguous buckets, e.g., 0-44 bytes, 45-90 bytes, 91-180 bytes, 181-260 bytes, 251-576 bytes, 577-1120 bytes, and larger than 1120 bytes), we experimented with different bucket widths as we created histograms of troop sequence lengths. In figure 9, we bucketed packet sizes and analyzed sequence length from a five-minute trace on a domestic iMCI OC-3 link from November 1997.

The figures, one for each direction of traffic on the link, both indicate the presence of a considerable number of trains of packets of sizes in the 261-576 byte range. Of interest to router engineers are the long trains of very short packets, i.e., the top red line on each graph in figure 6 represents trains of packets all of which were 44 bytes or less. Because a larger proportion of router processing overhead is per-packet than per-byte, sustained streams of relatively short packets pose a heavier workload on the router than when longer packets are interspersed.

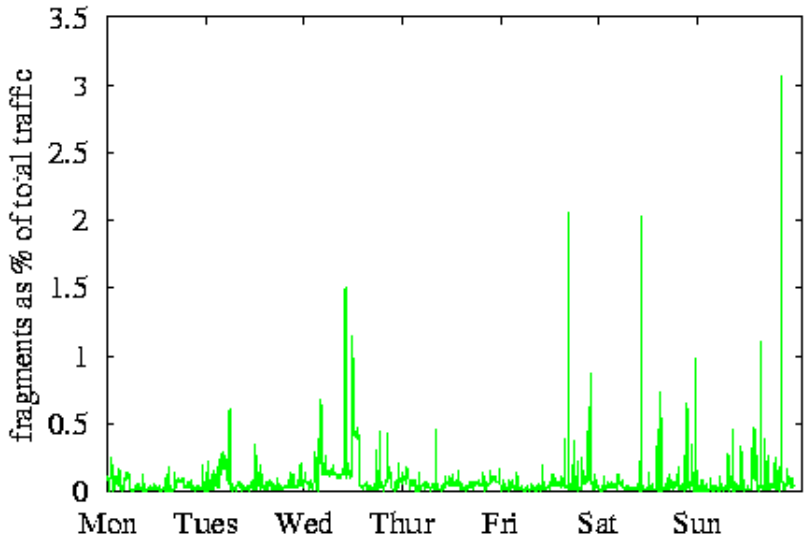
figure 9: lengths of sequences of packets of approximately same size.
(0.97 million packets in forward direction (left graph); 1.14m in other (right graph).
(5 minutes, domestic link, 5 November 1997).



IP fragment counts

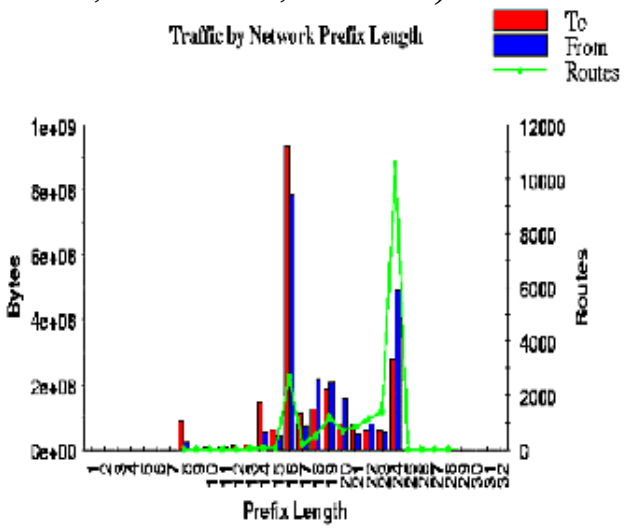
Several denial-of-service attacks exploit the presence of IP fragments, often found in a heterogeneous wide-area traffic stream [7], and the presence of IP fragments has thus become of concern for Internet service providers. Figure 10 illustrates the percent of total traffic, that is composed of fragmented IP packets over the course of a week in April 1998, at most 3% of total traffic.

figure 10: percent of traffic composed of fragmented IP packets *(13-20 April 1998)*



IP address prefix length distribution

figure 11: traffic sourced from and destined to addresses of various prefix lengths (packets and bytes)
(5 minutes, domestic link, 5 dec 1997).



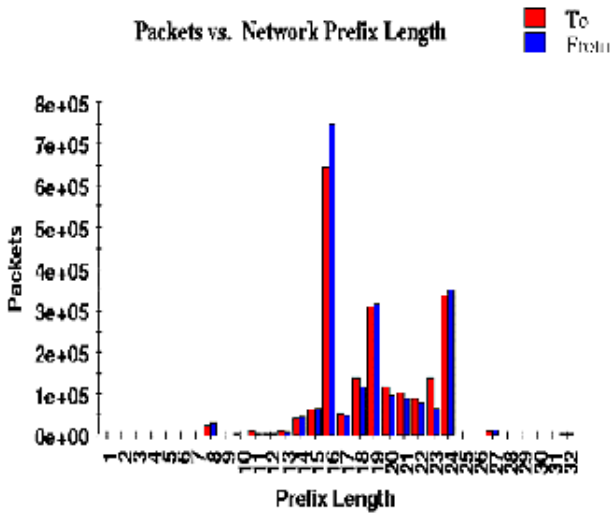


Figure 11 plots the distribution of prefix lengths for source and destination IP addresses of traffic in a five-minute packet trace from a domestic OC-3 iMCI trunk in December 1997. Most of the IP traffic, is sourced from and destined to IP addresses of prefix length /16, /24, and /19's, approximately 64.7% and 62.9% of the bytes, respectively, and 67.9% and 61.8% of the packets, respectively. For IP destinations, 30.9% of the packets and 36.8% of the bytes are destined to /16's, 16.2% of the packets and 13.4% of the bytes are destined to /24's, and 14.7% of the packets and 12.7% of the bytes are destined to /19's. For IP sources 35.9% of the packets and 26.5% of the bytes are sourced from /16's, 16.8% of the packets and 20.8% of the bytes are sourced from /24's, and 15.2% of the packets and 17.4% of the bytes are sourced from /19's. In contrast, the distribution of routing table entries comprises a larger proportion of /24's. Note that /19 is the minimum allocation by the registries of provider-independent address space, and that some providers are by policy refusing to forward or support route prefixes longer than /25.

figure 12: mean packet size as a function of prefix lengths
(5 minutes, domestic link, 5 dec 1997).

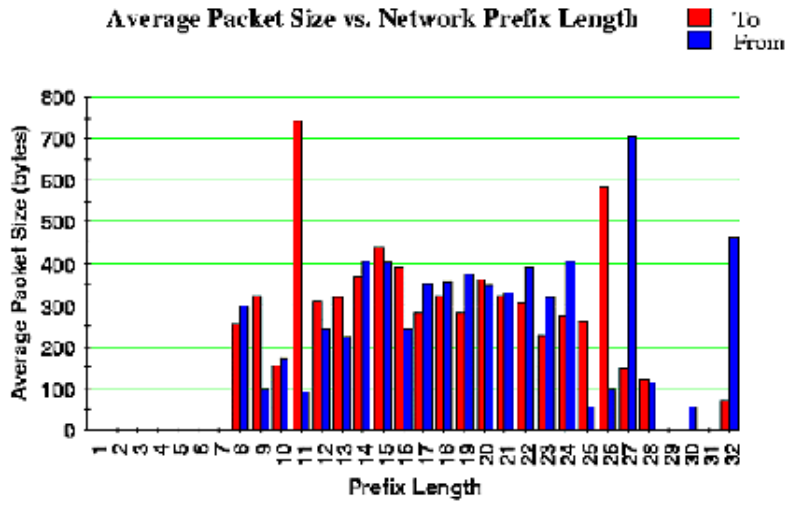


Figure 12 shows the average packet size as a function of prefix length. Note that in this data, prefix lengths of /24 and above tend to send larger packets, presumably to destinations of smaller prefix length, who are receiving packets of larger average length in bytes. The disparity between the average sizes of send and receive for /32 addresses may be due to a BGP route update from a router(s) announced as a /32.

Summary

We have presented statistics of recent iMCI Internet backbone data, taken in mid-April 1998, with some studies of packet traces from November and December 1997. The data was collected with Coral monitors developed by MCI under the auspices of the NSF-funded vBNS project [4] and in collaboration with NLANR/CAIDA [4,5]. The flow data is based on the monitors reporting on all expired flows, which are uni-directional, on back-to-back 5-minute intervals. We reported statistics on packet sizes, composition of traffic by protocol, per flow averages (overall and per application), traffic flow among countries, lengths of packet troops, IP fragment counts, IP address prefix length distribution, and IP address space utilization.

For the 24 hour period of 13 April 1998, a domestic iMCI backbone link saw 2.3 billion total packets and .8 trillion bytes, for which the top ten TCP flows constituted 122 million flows, 1.867 billion packets and 708.440 billion bytes. UDP traffic constituted 17 million flows, .1 billion packets, and 14 billion bytes.

Our measurements from April 1998 indicate that almost 60% of packets are 44 bytes or less, constituting a total of 7% of the byte volume, over half of the bytes are carried in packets of size 1500 bytes or larger.

The composition of traffic, in terms of both packets and bytes, is not significantly different from traces measured from last August 1997. TCP still averages about 95% of the bytes, 90% of the packets, and 80% of the flows. UDP makes up most of the rest of the traffic, with IPv6, encapsulated IP (IP-in-IP), ICMP, and other protocols taking up around 3% of the traffic. HTTP is still the dominant application on this backbone link, comprising up to 75% of the bytes, 70% of the packets, and 75% of the flows when client and server traffic are considered together. There is a significant amount of traffic in the *‘other’* category, spread among a wide range of TCP and UDP port numbers, many of which are also web-related (e.g., port 8080) Notably, the telnet protocol accounts for about 1% of the packets, and less than 1% of the bytes and flows, a marked decrease from recent years as alternative interactive protocols (e.g., ssh, kerberos, rlogin) have increased in popularity.

The distribution of flow sizes, as measured in packets, is long-tailed. Our measurements indicate that the majority of flows are still transaction-style, e.g., HTTP, SMTP, DNS, carrying much less traffic than the bulk data transfer-style flows, e.g., ftp-data, nntp.

We analyzed the length of packet trains composed entirely of packets of similar size. Because a larger proportion of router processing overhead is per-packet than per-byte, sustained streams of relatively short packets pose a heavier workload on the router than when longer packets are interspersed. These figures are thus important for router designers and engineers to frame their design specifications.

The percent of total traffic that is composed of fragmented IP packets over the course of a week in April

1998, at most 3% of total traffic.

We also analyzed the distribution of traffic by IP address prefix length, and found that the majority is sourced from and destined to IP addresses of prefix length /16, /24, and /19's, approximately 64.6% and 63.1%, respectively for sources and destinations.

Many of these statistics are of significant interest to Internet engineers and equipment vendors trying to design routing and switching equipment based on current workload profiles predicted trends. We expect such data will become only more important in the future as the Internet relentlessly evolves to higher speeds and more heterogeneous workloads.

Future work

Future possible avenues for Coral development/deployment include

1. expand the family of Coral monitors to address different speeds (OC12, OC48, DS3), interface types (e.g., Digital gigaswitch), OSes (e.g., UNIX), and other requirements (e.g., IP over Sonet).
2. more extensive performance testing to see when the boxes begin to fail to keep up with the traffic load.
3. sustainable security, privacy, and trust model
4. enhancements for remote access, including support for automatic rebooting, remote interactive access via terminal servers, and automatic scheduled data collection in various customizable modes
5. simplified ('plug-and-play') installation and maintenance by the local site to a single box only, with no miscellaneous equipment or phone lines
6. functionality to support collection of more detailed distribution information

Acknowledgements

The authors are grateful to Hans-Werner Braun for his commitment to Coral data analysis, Tracie Monk for her commitment to CAIDA, Rick Wilder for his support of the Internet measurement agenda, Nancy Bachman, Daniel McRobb, and David Moore, from CAIDA, and Renato Gragasin from MCI for their swat team efforts and proofreading and feedback on visualization impact. And none of this analysis would have been possible without the incredible effort of Joel Apisdorf in developing and maintaining the Coral monitor.

This work has been supported through MCI and the vBNS project, sponsored by NSF grant NCR-9321047, and through CAIDA, funded partially by the National Science Foundation through grant #NCR-9711092 but mostly through contributions from its industrial membership.

References

1. **OC3mon: Flexible, Affordable, High-Performance Statistics Collection** J. Apisdorf, k claffy, K. Thompson, and R. Wilder, http://www.isoc.org/isoc/whatis/conferences/inet/97/proceedings/F1/F1_2.HTM

2. **Wide Area Internet Traffic Patterns and Characteristics** K. Thompson, G. Miller, and R. Wilder, IEEE Network, Nov 1997. http://www.vbns.net/presentations/papers/MCItraffic.ps
3. **Coral home page** http://www.caida.org/Tools/Coral/
4. **very High performance Backbone Service** http://www.vbns.net
5. **Cooperative Association for Internet Data Analysis** http://www.caida.org
6. **National Laboratory for Applied Network Research (NLANR)/ Measurement and Operations Analysis Team (MOAT)** http://moat.nlanr.net/
7. **''Security Considerations - IP Fragment Filtering'',** G. Ziemba, D. Reed & P. Traina, RFC 1858, http://ds.internic.net/rfc/rfc1858.txt
8. **''Long-term traffic aspects of the NSFNET'',** K. Claffy and H.-W. Braun and G. Polyzos, Proceedings of INET'93. http://www.caida.org/Papers/lta.html
9. **A parameterizable methodology for Internet traffic flow profiling**, , K. C. Claffy, Hans-Werner Braun, George C. Polyzos, IEEE JSAC April 1996, http://www.caida.org/Papers/pmi.html
10. **tcpdump**, Lawrence Berkeley Laboratory Network Research Group, http://www-nrg.ee.lbl.gov/tcpdump.tar.Z
11. **TCP/IP Illustrated, Volume 1: The Protocols** W. Richard Stevens, Addison-Wesley, 1994.

Biographies

Gregory J. Miller has been a Senior Engineer in the vBNS Engineering Group at MCI since September 1996. His focus is on network performance measurement, traffic analysis, and IP and ATM Quality of Service mechanisms. Before joining MCI, he was a Senior Member of the Technical Staff at the MITRE Corporation. He received a B.S. degree from Loyola College in 1988, and the M.S. and Ph.D. degrees from the University of Delaware in 1990 and 1993, all in computer science.

Kevin Thompson is a Senior Engineer in the vBNS Engineering Group at MCI. He supports statistics collection architecture and implementation for the vBNS. He was employed as an engineer at the MITRE Corporation in the Networking Center until 1995. He received a B.S. in Computer Science from the University of Virginia in 1987 and an M.S. in Computer Science from the George Washington University in 1992.

k claffy is principal investigator for the Cooperative Association for Internet Data Analysis (CAIDA), and resident research scientist based at the University of California, San Diego. kc's research interests include traffic analysis, impact of high-demand (e.g., multimedia) applications on the integrity of current infrastructure, equity among users, and changing financial structure of the Internet.

---

23 apr 1998
kc@caida.org