LAN 패킷 분석을 통한 WWW이 통신상에 끼치는 영향

Ben Lee† · Andreas Schmid† · 고 진 광†† · 곽 한 탁†

요 약

1990년 초기에 알려지기 시작한 WWW은 그 이용량의 급격한 성장으로 LAN의 패킷 구성비율 및 통신량에 큰 변화를 가져오게 되었다. 이에 본 논문은 전형적인 학술기관의 LAN 이용분포를 분석하였다. 이를 위하여 HP LAN Protocol Analyzer와 tcpdump 프로그램등을 사용하였으며 LAN 트래픽과 네트워크 프로토콜의 상세한 분석에도 중점을 두었다. 특히 어떤 네트워크 프로토콜과 네트워크 패킷이 주로 사용되고 있는지 그 종류와 구성비율, 크기 등을 연구하였다. 또한 WWW가 전체 LAN 사용량 중 어떤 비중을 차지하고 있으며, 주로 어떤 MIME형태의 데이터가 사용되고 있는지도 연구하였다. 본 연구 결과에 따르면, 전체적으로 학술기관의 LAN은 여전히 메인 컴퓨터의 파일 시스템을 위한 NFS 프로토콜이 여전히 가장 큰 비중을 차지하고 있으며 WWW의 이용률은 예상만큼 높지 않았다. 한편 대부분의 네트워크 패킷은 실 데이터가 포함되지 않거나 크기가 작은 경우, 그리고 MTU에 달하는 크기의 패킷인 경우가 대부분을 차지하였다.

LAN Packet Trace Analysis-What is the Extent of WWW Traffic?

Ben Lee† · Andreas Schmid† · Jin Gwang Koh†† · Han-Tak Kwak†

ABSTRACT

Since its introduction in the early 1990s, the quick growth of the World Wide Web (WWW) traffic raises the question whether past LAN packet traces still reflect the current situation or whether they have become obsolete. For this study, several LAN packet traces were obtained by monitoring the LAN of a typical academic environment. The tools for monitoring the network were a stand-alone HP LAN Protocol Analyzer as well as the free-ware software tool tcpdump. Our main focus was placed on acquiring a low-level overview of the LAN traffic. Thus, we could determine what protocols were mainly used and how the packet sizes were distributed. In particular, we were interested in establishing the amount of WWW traffic on the LAN, and what MIME-Types this traffic is subdivided into. Our results indicate that in a typical academic environment such as ours, conventional sources of LAN traffic such as NFS are still predominant, whereas WWW traffic plays a rather marginal role. Furthermore, we verified that a large portion of the network packets contains little or no data at all, while another significant portion of the packets has sizes around the MTU. Consequently, research in the networking field has to direct its focus on issues beyond the WWW.

※ 이 논문은 1997학년도 순천대학교 교비 해외파견 지원에 의하여 연구되었음.
† 비 회 원 : 오리건 주립대학교 컴퓨터공학과
†† 정 회 원 : 순천대학교 컴퓨터과학과 교수
논문접수 : 1998년 8월 5일, 심사완료 : 1999년 9월 20일

1. Introduction

Starting out with the invention of the popular LAN Ethernet at Xerox PARC in the early 1970s, over the development of the internetworking protocols IP, TCP, and UDP around 1980, and culminating with the introduction of the WWW in the beginning of the 1990s, networking has become one of the most prominent paradigms of our time. Statistics[8] suggest that the internetworking (i.e., Internet) and the WWW have experienced an exponential growth since their inception. The quick spread of the usage of networking therefore poses a major challenge to the research in the area. Consequently, research efforts have targeted the physical transmission lines, the interfaces between computers and networks as well as the networking protocols and their implementation. As a result, the bandwidth of networks has increased tremendously from a few Kb/s to several Gb/s. Network interfaces have also become faster, and more efficient protocol implementations have reduced the processing time for transmission and reception of network messages.

This paper targets these issues. Inspired by the catchword "multimedia", the main goals of the paper, are to investigate the extent of LAN traffic on typical academic and office environments, such as ours, that can be classified as multimedia data, and how to possibly improve the network performance for this kind of traffic. Furthermore, we wanted to examine whether previous studies concerning non-HTTP traffic were still up-to-date.

The paper is organized as follows: Section 2 highlights some of the research efforts in the area of networking. The devices and tools that were employed to monitor a departmental network are presented in Section 3. The results of the performed network measurements are presented in Section 4. Finally, Section 5. provides a brief conclusion and addresses the future work to be done in the area.

2. Background and Related Work

Research in the area of networking concentrates on several different topics. First, there is a Network Interface (NI). At the University of Wisconsin-Madison, Mukherjee *et al.* performed a survey on the current NIs[14] and provided arguments for placing the NI not on the I/O bus, but on the memory bus [15]. They argued that the tighter integration between the processor and the NI offers multiple possibilities to increase the network performance. Other research efforts have been directed at the development of custom networks for high-speed parallel computing. The StarT-Voyager Cluster area network, developed at MIT, is such an example [1]. Each StarT-Voyager node is a commercial PowerPC 604-based SMP where one processor card is substituted by a *Network Endpoint Subsystem* (NES) card. Thus, the NES with its embedded *service processor* (sP) is located on the *application processor's* (aP's) cache-coherent memory bus. The sP consists of a PowerPC 604 processor, a memory controller and a DRAM. Its task is basically to service remote memory references of the aP and incoming memory requests from the network.

In the TCP/IP domain, research aims to improve the protocol processing time. Jacobson *et al.* [4, 7] studied the TCP processing overhead and concluded that the data-touching processing overhead could be reduced by limiting the number of data copies to one (referred to as *single-copy-* or *zero-copy-implementations*), and by incorporating the calculation of the Internet checksum into the data copy[19]. summarizes further TCP/IP improvements. First, it suggests the use of caches for the Protocol Control Block (PCB) lookup. For each TCP/IP connection, a PCB data structure is used to store the IP addresses and port numbers of both connection endpoints. Since the corresponding PCB for each received message has to be found, the employment of caches would be truly helpful.

In the area of Ethernet traffic traces, Mogul studied the locality of LAN packets at the scale of processes [16], thereby focusing on the IP protocol together with

TCP and UDP. Mogul also argued for the idea of a "persistent-connection" HTTP (P-HTTP)[17]. P-HTTP avoids the practice of HTTP/1.0 [3] of using separate TCP connections between WWW client and server to transfer the HTML code as well as each inlined image of a webpage. Consequently, this idea has been added to HTTP in the latest HTTP/1.1 Proposed Standard [6]. Finally, Mogul [18] evaluated the behavior of a LAN containing a busy WWW server. This study comes closest to ours, since Mogul also uses the tcpdump tool to monitor the network. However, the environments of a busy WWW server LAN and a typical academic or office LAN can hardly be compared especially since his findings concerning the packet size distribution were quite different from ours.

Last but not least, Kay and Pasquale [10, 11] evaluated two network traffic traces, one from a LAN, the other from a WAN. They found that on the LAN the median message sizes for TCP and UDP were 32 and 128 bytes, respectively. On the WAN, 99.7% of all messages were at most 500 bytes long. In the presence of such small messages, the non-data-touching parts of the TCP/IP implementation become relevant. [10, 11] show profiles of the processing times for sending and receiving TCP and UDP messages over a range of message sizes. These profiles demonstrate that for small messages the non-data touching processing overhead actually dominates the overall processing cost. Moreover, the non-data-touching processing overhead consists of many different, independent operations, thus making speedups tedious or even impossible to achieve without giving up the present functionality.

3. Network Monitoring Setup

Our departmental network consists of about 300 nodes connected by the same 10Mb/s Ethernet. The nature of the nodes is diverse. It consists of HP workstations running HP-UX, Sun workstations running Solaris, Intel machines running either Linux or Windows NT, and NeXT computers. The tools used for monitoring the network were the HP LAN Protocol Analyzer and the software tools that run on HPUX workstations. As suitable software tools, nettl, developed by HP, and tcpdump by the Lawrence Berkeley National Laboratory (LBNL) were at our disposal. For reasons of practicality we chose the latter, tcpdump is the most widely used packet-capturing tool, probably due to its platform independence and free availability. In addition, there exist some applications that digest tcpdump's output, such as tcptrace and xplot.

The HP 4972A LAN Protocol Analyzer, herein referred to simply as the "Analyzer", is a stand-alone device that is directly connected to the network through its transceiver interface.

tcpdump [9] was developed by members of the Network Research Group (NRG) of the Information and Computing Sciences Division (ICSD) at Lawrence Berkeley National Laboratory (LBNL) in Berkeley, California. It serves as a LAN packet-capturing tool and uses the library libpcap[1], also written by NRG.

Since all available programs that digest tcpdump data merely focus on TCP/IP traffic, a C program called snarf was developed. It collects the desired statistics from a tcpdump parsed data output. The collected statistics include the packet and byte counts for all known protocols, e.g., Appletalk, the Internet Control Message Protocol (ICMP), the Address Resolution Protocol (ARP), as well as all TCP and UDP ports that appeared in packet headers along with the number of data bytes sent to and from them. In addition, the overall throughput in bytes per second is calculated, and eventually the distribution of the data byte sizes found in the packets.

Finally, tcptrace is one of the programs that is freely available[2] which reduce the tcpdump data. Unlike snarf, tcptrace exclusively examines the TCP/IP packets in the tcpdump raw data tracefile, including statistics of all HTTP connections found in the file. Nevertheless, the functionality of the HTTP

1) tcpdump and libpcap are available at http://ee.lbl.gov/
2) Available at http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html

module was not sufficient as it failed to detect the content type of the data, let alone statistics about how many packets and data bytes per content type were traced. Consequently, these features were added by the authors.

4. Monitored Results

The various network monitoring tools described in the previous section produced a respectable amount of data. These results will be presented in this section. We will start out with the results of the Analyzer in Section 4.1, then continue with the presentation of an unfiltered trace of tcpdump in Section 4.2. Section 4.3. We will show the tcpdump traces with HTTP filter, and all these results will be summarized in Section 4.4.

4.1 HP LAN Protocol Analyzer Results

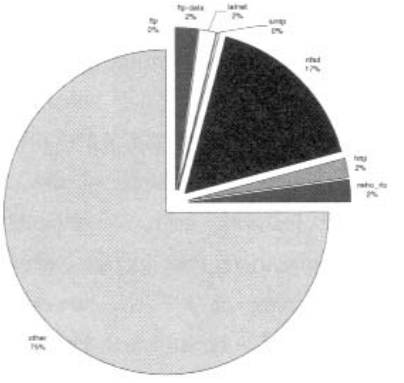
The Analyzer measurements were performed by choosing a 24-hour measurement time and a 144 min. Sample time, thus yielding ten samples and a ten-sample average value over an entire day. The days of the measurements were chosen arbitrarily seven in total. The results are listed in <Table 1>. Additionally, for illustration purposes, (Figure 1) shows the same data depicted in a pie chart.

<Table 1> Seven-Day Summary of TCP/UDP Port Activity

TCP/UDP Port	Percentage	Frames
Ftp	0.091570	2688590
Ftp-data	2.043720	60005650
Telnet	1.857451	54536630
Smtp	0.035122	1031210
Nfsd	16.934220	4.97E+08
Http	1.849966	54316860
Rwho_rlo	2.124210	62368930
Other	75.065600	2.204E+09
Total data bytes	1.17E+10	
Total frames	2.94E+07	

We notice that of seven detailed ports, the Network File System (NFS) port was the most frequently used. 16% of all frames had this port as either the source or the destination address. Four of the remaining ports were referenced by circa 2% of all frames each. They are the File Transfer Protocol data (ftp-data) port, the Remote Terminal (telnet) port, the HTTP (http) port, and the Remote Login (rwho_rlo) port. By considering the total data bytes and total frame number, the average data content of a frame is calculated to be 398 Bytes. Furthermore, the average throughput over the whole seven days was 19,345 Bytes/sec.

Despite its limitations, the Analyzer was still valuable as a measurement tool that works independently from the software tools whose monitoring results will be discussed later in this chapter. The results of both the Analyzer and the software tools will be compared in Section 5.



(Figure 1) Seven Day Summary of TCP/UDP Port Activity

4.2 Unfiltered tcpdump Traces

Several unfiltered LAN traces were collected in order to determine what protocols were used across the network and how much they contributed to the overall network traffic. Another objective of these traces was to further detail the measurements of the Analyzer in terms of the TCP/UDP port activity. The trace file to be presented here is the file mar09out4. <Table 2> outlines its basic characteristics The length of the trace is 4000,000 packets. Such a lengths was the result of the observation time of 43 min. 34 sec. Recording this particular trace took the usage of the system resources to its upper limit. The size of the

raw data output file was 58 Mbytes, and the output file after the parsing step of tcpdump still had a respectable size of 27Mbytes.

〈Table 2〉 Unfiltered Trace mar09out4 of 400,000 Packets

File	Mar09out4
Start of measurement	Mon Mar 9, 17:59:51 1998
End of measurement	Mon Mar 9, 18:43:25 1998
Elapsed time	2614.323311 secs
Total number of frames	400000
Total Bytes on Net	126862520

<Table 3> shows the various types of protocols the traced packets used. As can be seen, the majority of the frames, pecifically 58%, were UDP frames. They carried an overwhelming 91% of all data bytes. TCP came in second with 32% of all frames, yet only accounted for 3.5% of the total data bytes on the network, which yields an average of merely 34 Bytes per frame. The reason for UDP being more popular than TCP lies in the fact that UDP offers a faster protocol processing. This comes at the price of the lack of TCP features, such as reliability or flow control. For applications that operate exclusively in the trusted environment of a LAN, however, the UDP services are deemed to be sufficient. Despite contributing only 3% of all frames. Ethernet broadcast messages are still responsible for 4% of the network data traffic, due to an average of 415 Bytes per frame. They represent various services, such as EtherTalk. The third largest proportion in terms of frames is the Address Resolution Protocol (ARP), which is used to determine the corresponding Ethernet address of an IP address. It represents 7% of all network frames, suggesting that a lot of network traffic had either a source or a destination host outside the LAN. The remaining protocols are: the Internet Group Management Protocol (IGMP), which used to manage multicast group; the Internet Control Message Protocol (ICMP) that communicates error messages on the Internet; and Open Shortest Path First (OSPF), which is an interior gateway protocol.

<Table 4> lists a more detailed observation of those TCP ports that are "well-known" and appeared as either the source or the destination port in TCP frames. "Well-known" means that these ports are listed in the (in our environment HP-UX) file /etc/services and have a certain kernel space service associated with them. This service is indicated by their official service name. All ports that are not well-known generally belong to user-space applications. We find that 51% of the TCP frames in <Table 4> belong to telnet traffic, followed by 38% for ftp-data. Since the telnet data per frame is only a single byte on average, telnet accounts only for 2% of the data flow of the well-known TCP ports. Considering the data byte proportions it can be observed that HTTP and therefore WWW usage is responsible for 63% of this table's data byte traffic. Since this amount of data

〈Table 3〉 Protocol Distribution of unfiltered Trace mar09out4.

Protocol	Frames	Bytes	Average Bytes/Frame	Bytes/Total Bytes in %	
Appletalk	327	149143	456	0.118	
ARP	29372	822416	28	0.648	
IGMP	183	5124	28	0.004	
ICMP	11	308	28	0.000	
OSPF	262	11528	44	0.009	
Ethernet broadcast	12150	5036932	415	3.970	
Other Ethernet	1874	109839	59	0.087	
	Frames	Data Bytes	Average Data Bytes/Frame	Data Bytes/Total Bytes in %	Total Bytes TCP
TCP	127896	4409311	34	3.476	9525151
	Frames	Data Bytes	Average Data Bytes/Frame	Data Bytes/Total Bytes in %	Total Bytes UDP
UDP	227919	116317609	510	91.688	121089817

〈Table 4〉 Well-known TCP Ports appearing in Trace mar09out4.

TCP Ports					
Port Name	Port Number	Frames	Data Bytes	Average Data Bytes/Frame	Data Bytes/ Total TCP Data Bytes in %
Ftp-data	20	17987	510346	28	11.574
Ftp	21	439	5141	12	0.117
Telnet	23	23859	27235	1	0.618
Smtp	25	245	5558	23	0.126
Http	80	1110	954471	860	21.647
Pop3	110	541	2595	5	0.059
Portmap	111	32	0	0	0.000
Nntp	119	143	18972	133	0.430
Login	513	2534	3918	2	0.089
Printer	515	70	20	0	0.000
Msql	1111	497	259	1	0.006
Nfsd	2049	142	1687	12	0.038

resides only in 2% of all frames, this yields a high ratio of 860 bytes/frame. As to the corresponding UDP table, shown in <Table 5>, it is obvious that the frames created by the NFS (nfsd), which provides remote

〈Table 5〉 Well known UDP Ports appearing in Trace mar09out4.

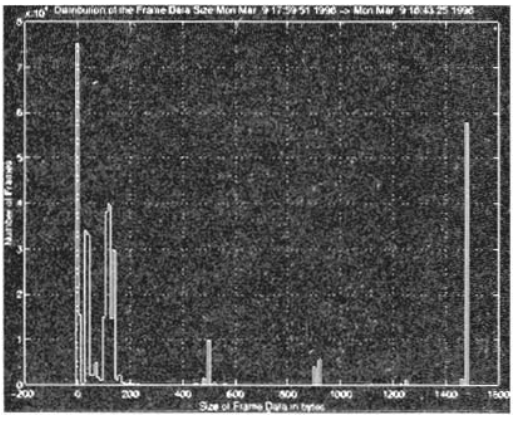
UDP Ports					
Port Name	Port Number	Frames	Data Bytes	Average Data Bytes/Frame	Data Bytes/ Total UDP Data Bytes in %
Tcpmux	1	12954	2010212	155	1.728
Domain	53	4333	279910	65	0.241
Bootps	67	35	10500	300	0.009
Bootpc	68	35	10500	300	0.009
Portmap	111	402	19512	49	0.017
Netbios_ns	137	1046	55936	53	0.048
Netbios_dgm	138	270	57310	212	0.049
Snmp	161	337	25989	77	0.022
Biff	512	8	83	10	0.000
Login	513	262	99720	381	0.086
Krbupdate	760	977	148464	152	0.128
DAServer	987	1	124	124	0.000
Nfsd-keepalive	1110	2	54	27	0.000
Msql	1111	2	112	56	0.000
Rlb	1260	1	36	36	0.000
Clvm-cfg	1476	1	90	90	0.000
Ingreslock	1524	1	32	32	0.000
Nft	1536	1	31	31	0.000
Sna-cs	1553	1	22	22	0.000
Ncpm-pm	1591	1	128	128	0.000
Ncpm-hip	1683	1	121	121	0.000
Cvmon	1686	1	36	36	0.000
Pmlockd	1889	2	177	88	0.000
Nfsd	2049	212623	114362992	538	98.320
Eklogin	2105	1	38	38	0.000
Netdist	2106	1	33	33	0.000
Rfa	4672	2	110	55	0.000
Veesm	4789	2	67	34	0.000

access to shared files across networks, by far outnumber all others. They represent 91% of all frames and hold 98% of all data bytes. In second place follows the tcpmux service, more than an order of magnitude smaller in number, both for frames and sizesbelow 200 bytes and at 1,480 bytes. The occurrence of many small messages is striking. Specifically, 78% of all messages contain no more than 200 data bytes. This indicates that a huge bulk of messages simply consists of control messages for data being transferred in the other direction of the connection. On the other hand, roughly 15% of all frames carry 1,480 byte data. These frames can be attributed to large UDP messages that have been split up into fragments based on the Maximum Transfer Unit (MTU) size plus one fragment for the remainder of the message.

Finally, (Figure 3) shows the throughput graph. The throughput graph is admittedly somewhat densely packed, but shall be presented anyway for completeness.

The bursty nature of the LAN traffic is nevertheless still evident, with peaks reaching up to 730 Kbytes/sec.

Another interesting characteristic of the observed LAN is the distribution of data field sizes for the frames. (Figure 2) shows this distribution for the mar09out4 trace in a resolution of 10 bytes. It is immediately evident that there are two diametrically



(Figure 2) Distribution of Frame Data Sizes for mar09out4

opposed accumulations of data bytes. tcpmux [12] is an internet standard that can be used by future TCP services instead of using "well- known" ports.

4.3 tcpdump Trace with HTTP Filter

After capturing the unfiltered tcpdump traces, several traces were gathered with a filter so that only frames having TCP/UDP port 80 as a source or a destination port were recorded. The raw tcpdump traces were written to output files, which in turn were used as input files for the tcptrace tool.

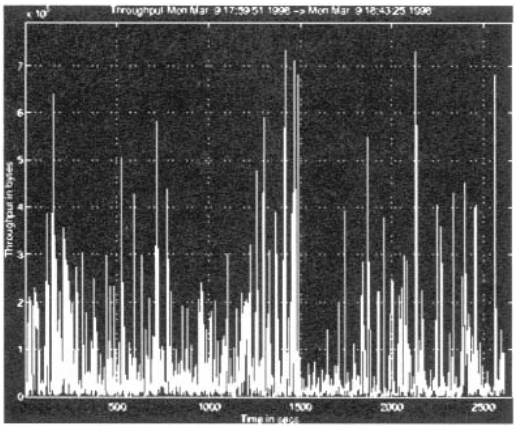
In the course of monitoring the HTTP traffic, it soon became clear that the results obtained in Subsection 4.1 of only 2% HTTP frames on the LAN were indeed realistic. The largest trace that was collected with HTTP Filter, mar11out5, is composed of 60,000 frames, and it took more than 19 hours <Table 6> to obtain it.

<Table 6> Trace mar11out5 with HTTP Filter

File	mar11out5
Start of measurement	Wed Mar 11, 23:15:52 1998
End of measurement	Thu Mar 12, 18:22:04 1998
Elapsed time	68771.354752 secs
Total number of frames	60000
Total Bytes on Net	33448902

In (Figure 4), the distribution of the frame data sizes is depicted. The highest peak with 29,914 frames or 49.9% of all frames is distinctly at zero bytes, so consequently every other TCP packet had no data at all, but just a header with flags such as an ACK or a SYN. Another high peak, as seen in the unfiltered trace, is close to the MTU size of 1,460 Bytes, where 32% of all frames reside. Note that this time the maximum data size is 20 bytes less than in the previous case, due to the fact that the TCP header takes away 20 bytes as compared to the 8 header bytes for UDP. Eventually, 4.2% or 2,500 frames had a data size of 520 bytes, and 1% of the frames carried

590 bytes. Hence, 87% of all frames have only one of four distinct sizes.



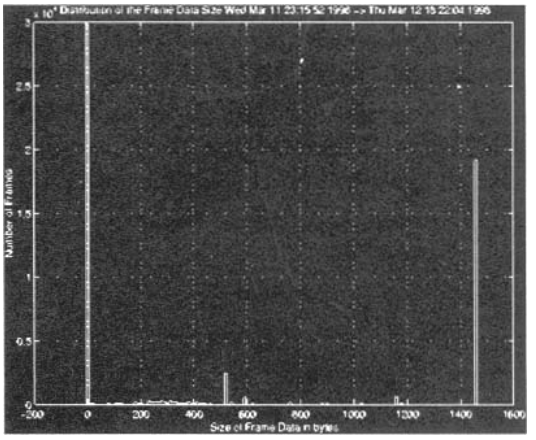
(Figure 3) Throughput for mar09out4

Such a behavior of the frame size distribution was not unexpected. Instead, it is typical of HTTP/1.0 [3], which is currently used by WWW clients and servers. In this version, the HTML text of a requested webpage is first transferred. By parsing this HTML document, the WWW client then determines all the inlined elements, such as images or sound files. For each inlined element to be transferred, the client opens a separate TCP connection. Thus, a lot of overhead in the form of TCP control messages is created, which explains the large percentage of messages (49.8%) that transported no data at all. The newer HTTP/1.1 Proposed Standard [6] alleviates this problem by adding the concept of "persistent TCP connections" [17]. This implies that only one TCP connection is established between the client and the server to transfer all subsequent HTTP requests and their replies. Nevertheless, HTTP/1.1 has not yet been put to use in practice.

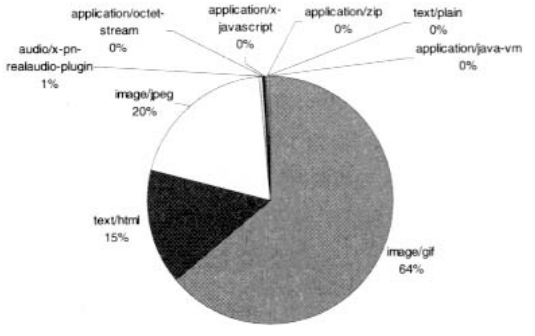
In addition to the information collected by snarf, the results of tcptrace come into play. They are shown in <Table 7> in tabular form and graphically in (Figure 5). The most requested Multi-purpose Internet Mail Extension (MIME)-type was image/gif representing 64% of all requests, whereas image/jpeg comes in second with 20%, followed by text/html with 15%. Eventually, audio/x-pn-real audio-plugin accounts for 1% of the recorded requests, where real audio is the de-facto audio streaming standard on the Internet. All other types play only a marginal role. When considering the amount of data bytes sent, the picture looks somewhat different, which is a phenomenon that is mostly due to eight tremendously large requests for the type application/octet-stream. Denoting either binary files or files with unknown MIME-Type, the type application/octet-stream is used for binary and general file transfers via the WWW Browser. Therefore, the average requested file size of more than 11 MBytes is huge, but realistic. One order of magnitude smaller, the three MIME-Types image/jpeg, image/gif and text/html ensued.

<Table 7> Content-Types of HTTP Requests in mar11out5.

Content-Type	Requests	Bytes	Bytes/Request	Bytes/Total Bytes
Image/gif	1283	5022950	3915	4.812
Text/html	304	1664316	5475	1.595
Image/jpeg	397	6552625	16505	6.278
Audio/x-pn-realaudio-plugin	11	176	16	0.000
Application/octet-stream	8	91073218	11384152	87.254
Application/x-javascript	2	15809	7904	0.015
Application/zip	1	16399	16399	0.016
Text/plain	5	26459	5292	0.025
Application/java-vm	1	4807	4807	0.005
Total	2012	104376759	51877	100.000



(Figure 4) Distribution of Frame Data Sizes for mar11out5.



(Figure 5) Shares of the different Content-Type Requests.

5. Conclusion and Future Work

We started out with the goal of investigating the extent of traffic on the departmental network that can be classified as multimedia data, and how to possibly improve the network performance for this kind of traffic. In Section 4, the monitored results were presented in detail. The conclusions that can be drawn from these results are as follows:

As for the characteristics of the unfiltered trace, our results correspond very well with the results found by [11], who examined a LAN trace from 1992. The monitored LAN traffic still featured a bimodal characteristic, where most of the frames either have very small data sizes or data sizes near the MTU. The dominant protocol was undisputedly UDP, transporting mostly NFS services.

It turned out that HTTP traffic constitutes only a very small fraction of the overall departmental LAN traffic. According to the Analyzer's seven-day average, this fraction amounts to only 1.85% of all frames. In the 43 min. 34 sec. trace mar09out4, merely 0.28% of all recorded frames were determined to be HTTP frames, carrying 0.75% of all data bytes. Thus, it can be stated that the emergence of the WWW may not have yet brought about a significant change in the utilization of a typical academic or office LAN. Additionally, the HTTP traffic still features a bimodal characteristic of the overall traffic. As a consequence, speeding up multimedia network traffic implies speeding up all network traffic, and vice versa. Therefore, the fraction of messages that have large data sizes does not pose any greater problem. More important problem to deal with are the small-sized messages, specifically those with null data bytes.

In order to decrease the number of HTTP-induced TCP messages, it is necessary that the improved HTTP/1.1 Proposed Standard is implemented. HTTP/1.1 will not be able to however alleviate the intrinsic problems of HTTP that are caused by the requestreply paradigm. Each request or reply forces the opposite communication endpoint to answer with a TCP ACK message. This ACK message does not contain any data because the protocol layering and the time to process the HTTP request or reply delay the generation of data. An elimination of such empty TCP control messages seems only feasible by collapsing the protocol layers of TCP and HTTP.

The variety of the Content-Types of HTTP messages is rather small. image/gif, image/jpeg and text/html are clearly predominant. This fact, together with the small share of HTTP LAN traffic, proves that the academic environment has lost its role as the driving force of the WWW. Responsibility for the rapid growth of the Web now belongs to the private and the business sectors. Therefore, improving the performance particularly on academic LANs is best

achieved by concentrating the research on the further development of the transport and the network layer protocols. This next generation of the Internet must not only provide an optimum efficiency, but qualitatively different services than the current Internet. The necessary services include the guarantee of quality-of-service (QoS) attributes. Specifically, the guarantee of the minimum bandwidth, the maxi mum latency, and an upper limit on packet loss is highly desirable. Other important aspects are to improve the data and network security as well as providing a better network scalability.

References

[1] Boon S. Ang, Derek Chiou, Larry Rudolph, and Arvind. Message Passing Support on StarT-Voyager. CSG-Memo-387, MIT Laboratory for Computer Science, July 1996.

[2] Martin F. Arlitt and Carey L. Williamson. Web Server Workload Characterization : The Search for Invariants. In *1996 ACM SIGMETRICS Conference*, May 1996.

[3] T. Berners-Lee, R. Fielding, H. Frystyk. *Hypertext Transfer Protocol HTTP/1.0*. Internet RFC 1945, May 1996

[4] David D. Clark, Van Jacobson, John Romkey, and Howard Salwen. An Analysis of TCP Processing Overhead. In *IEEE Communications Magazine*, pages 23-29, June 1989.

[5] Thorsten von Eicken, David E. Culler, Seth C. Goldstein, Klaus E. Schauser. Active Messages : a Mechanism for Integrated Communication and Computation. In *Proceedings of the 19th International Symposium on Computer Architecture*, May 1992.

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee. *Hypertext Transfer Protocol HTTP/1.1*. Internet RFC 2068, January 1997

[7] Van Jacobson. Some Design Issues for High-speed Networks. *Networkshop '93*, November 1993.

[8] Gregory R. Gromov. *History of Internet and WWW : The Roads and Crossroads of Internet's History*. Webpage at http : //www.internetvalley.com/intval.html

[9] Van Jacobson, Craig Leres, and Steve McCanne. *Tcpdump(1) dump traffic on a network*. Unix Manual Page, 1996.

[10] Jonathan Kay and Joseph Pasquale. The Importance of Non-Data Touching Processing Overheads in TCP/IP. In *SIGCOMM '93*, pages 259-268, 1993.

[11] Jonathan Kay and Joseph Pasquale. Profiling and Reducing Processing Overheads in TCP/IP. In *IEEE/ACM Transactions on Networking*, December 96

[12] M. Lottor. *TCP Port Service Multiplexer (TCPMUX)*. Internet RFC 1078, 1988.

[13] Alan M. Mainwaring and David E. Culler. Active Message Applications Programming Interface and Communication Subsystem Organization. U.C. Berkeley Technical Report #CSD-96-918, October 1996.

[14] Shubhendu S. Mukherjee and Mark Hill. A Survey of User-Level Network Interfaces for System Area Networks. Technical Report 1340, Computer Science Department, University of Wisconsin-Madison, February 1997.

[15] Shubhendu S. Mukherjee and Mark Hill. A Case for Making Network Interfaces Less Peripheral. In *Hot Interconnects V*, 1997.

[16] Jeffrey C. Mogul. Network Locality at the Scale of Processes. In *SIGCOMM '91*, pages 273-284, 1991.

[17] Jeffrey C. Mogul. Network Locality at the Scale of Processes. In *ACM SIGCOMM '95*, pages 299-313, August 1995.

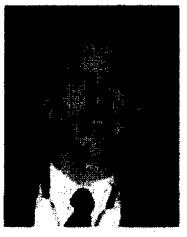
[18] Jeffrey C. Mogul. Network Behavior of a Busy Web Server and its Clients. Research Report 95/5, DEC Western Research Laboratory, October 1995.

[19] Craig Partridge. *Gigabit Networking*. Addison-Wesley, 1994.

Ben Lee

e-mail : benl@ece.orst.edu

Ben Lee received the B.E. degree in electrical engineering from the State University of New York (SUNY) at Stony Brook, New York, in 1984 and the Ph.D. degree in computer engineering from The Pennsylvania State University, University Park, in 1991. He is currently an associate professor in the Department of Electrical and Computer engineering at Oregon State University. His research interests include computer system architecture, multithreading and thread-level speculation, parallel and distributed systems, and program partitioning and scheduling. Dr. Lee is a member of the IEEE Computer Society.



Andreas Schmid

e-mail : schmidan@ece.orst.edu

Andreas Schmid received the B.S. degree in electrical engineering from University of Stuttgart University, Germany, and the M.S. degree in computer and electrical engineering at Oregon State University.

He has won the robot race at the IAS of University of Stuttgart, and interests in network computing, computer system and embeded controllers.

고 진 광

e-mail : kjg@sunchon.ac.kr

1982년 홍익대학교 공과대학 전자계산학과(이학사)

1984년 홍익대학교 대학원 전자계산학과(이학석사)

1997년 홍익대학교 대학원 전자계산학과(이학박사)

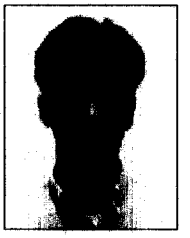
1984년~1988년 송원전문대학 전자계산과 전임강사

1988년~현재 순천대학교 자연과학대학 컴퓨터과학과 정교수

1993년~1994 홍익대학교 공과대학 컴퓨터공학과 국내교류교수

1997년~1998년 오리건주립대학교 컴퓨터공학과 방문교수

관심분야 : 데이터베이스, 트랜잭션 관리, 정보통신, CALS/EC 등



곽 한 탁

e-mail : hantak@ece.orst.edu

Hantak Kwak received the B.S. degree in electronic engineering from SungKyunKwan University, Korea, in1984, the M.S. degree in electrical engineering at South Dakota State University in 1987, and the Ph. D. degree in computer and electrical engineering from Oregon State University in 1998. His reseach areas are microarchitecure, parallel and distributed computing, SMP, and software and hardware multithreading.