Thresholds
Edited by Lawrence Bernstein

Measurement and Analysis of HTTP Traffic

Yogesh Bhole^{1,2} and Adrian Popescu¹

The usage of Internet is rapidly increasing and a large part of the Internet traffic is generated by the World Wide Web (WWW) and the associated protocol HyperText Transfer Protocol (HTTP). Several important parameters that affect the performance of the WWW are bandwidth, scalability, and latency. To tackle these parameters and to improve the overall performance of the system, it is important to understand and to characterize the application level characteristics. This article is reporting on the measurement and analysis of HTTP traffic collected on the student access network at the Blekinge Institute of Technology in Karlskrona, Sweden. The analysis is done on various HTTP traffic parameters, e.g., inter-session timings, inter-arrival timings, request message sizes, response code, and number of transactions. The reported results can be useful for building synthetic workloads for simulation and benchmarking purposes.

KEY WORDS: World Wide Web; HyperText Transfer Protocol; performance; service level agreement .

1. INTRODUCTION

The work on World Wide Web (WWW) started in 1989 with the development of a set of simple protocols and formats [1]. WWW has afterwards been more and more developed and also used as a test bed for sophisticated concepts for hyper-media and information retrieval. The consequence has been that WWW has had a major impact on modern life and influenced our lives in many ways. It has, for instance, become the preferred way of content distribution. Furthermore, the fact that the WWW technology is available on demand, made it very appealing to the user community. The Web technology has been quickly adapted for specific purposes like education, entertainment, and commercial environments (e.g., banking,

¹Department of Telecommunication Systems, School of Engineering, Blekinge Institute of Technology, Campus Gräsvik, Karlskrona, Sweden.

² To whom correspondence should be addressed 10, Suruchi, M. G. Road, Naupada, Thane(W.), Maharashtra 400602, India. E-mail: yogb21@yahoo.com.

shopping). As a consequence, researchers have been very active and did research to understand the complexity of WWW, to develop appropriate workload models for simulation and test bed purposes as well as to improve the performance. This is however a difficult task because the Web is actually consisting of a variety of software components (e.g., browsers, servers, proxies, back-end databases), and also due to the ever-changing characteristics of Internet traffic.

Today, some of the major challenges in WWW performance are scalability, latency, bandwidth and the problem of aborted connections. Related to this, it is important to understand and to characterize the application level characteristics. For instance, significant contributions have been made so far towards characterizing the application (e.g., client behavior, server behavior, proxy behavior, and structure of Web pages) [2, 3], towards better protocols [4] as well as for better caching strategies for clients and for servers [5]. These studies have revealed the complexity of WWW and the need for a deep understanding of WWW workloads. Furthermore, another important research is related to the development of generators of synthetic workloads, which has been a major focus of WWW research [6].

The purpose of this paper is to present a characterization study of traffic collected at the client side for the considered application, to be further used in a client-server simulation framework. Detailed results are reported on measuring, modeling, and analysis of client and server HTTP traffic collected from the student access network at the Blekinge Institute of Technology (BIT), Karlskrona, Sweden. In particular, we confirm the results of former studies showing that inter-session arrival times are exponential and that the number of transactions within a WWW session can be modeled by a negative binomial distribution.

Section 2 is devoted to reviewing some of the main solutions and challenges related to IP quality of service (IP QoS) with impact on the performance of WWW. Section 3 is about service level agreement (SLA) definitions and descriptions. Section 4 describes the measurement methodology used for the HTTP traffic. Section 5 reports the characteristics of HTTP traffic considered in the study as well as the summary statistics for the collected data sets. The paper is concluded in Section 6.

2. INTERNET QUALITY OF SERVICE

The new models put forth for IP QoS means that the focus of data networking has been shifted to data delivery with guaranteed delay performance (in the form of end-to-end delay and jitter). The new Internet is expected to provide services where several parameters (delay, jitter, and packet loss) are minimized as much as possible. New control mechanisms are under development, to help network operators providing reliable service levels and also differentiating versus competitors, where possible differentiators are metrics like connection setup and delay [7, 8].

Actually, the only one way to provide end-to-end delay guarantees is to create an end-to-end data flow and to reserve resources in the network. That

means that connection-oriented (CO) subnetworks must be used. Typical examples of CO subnetworks are asynchronous transfer mode (ATM), multiprotocol label switching (MPLS), frame relay (FR) as well as the integrated services (IntServ) model. The differentiating services (DiffServ) model is also typical for this case, except that performance guarantees can only be provided when the network is lightly loaded [9].

The ideal subnetwork for IP is however connectionless (CL). Two technologies have therefore been proposed for reducing the delay in this case. These are the header compression and using of fragmentation/reassembly [10, 11]. The area of applicability is however limited, e.g., header compression is used only for low-speed serial links.

Furthermore, due to the complexity of Internet traffic as well as of Internet protocols, spanning from medium access control (MAC) protocols at the link layer up to specific control mechanisms at the application layer, several other aspects have come to play an important role in the provision of end-to-end delay guarantees. These are the traffic self-similarity, the multilevel network control and the so-called (BGP) routing flaps [12].

Today, there is mounting evidence that traffic self-similarity is of fundamental importance for a number of traffic engineering problems, such as traffic measurements, queueing behavior and buffer sizing, admission control and congestion control [13]. Unlike traditional packet traffic models, which give rise to exponential tail behavior in queue size distributions and typically result in optimistic performance predictions and inadequate resource allocations, self-similar traffic models predict hyperbolic or Weibull (stretched exponential) queue size distributions and could therefore result in longer waiting times at the network processing elements (e.g., routers), affecting so the control and the management of the Internet [13].

Furthermore, some of the most serious impairments of the Internet, which could introduce delay variations too big to be compensated in the buffers at receivers, are traffic congestion (especially at the ingress routers) and routing flaps (at the core routers). The routing flaps may occur when changes in network routing occur, and they are like shock waves that propagate through the Internet's backbone. For instance, when a major core router, or a link, goes down, the other routers have to reconfigure the routing for the traffic addressed to the damaged router. Further, the information about the new (routing) configuration is also propagated to other routers, across multiple routing domains, to the farthest corners of the Internet. Should other changes in the Internet occur before the first change has completely propagated to the corners, a new set of ripples may appear that collide with the previous one, creating so a situation when routers are "flapping" instead of routing, and finally generating a continual background roar of changes in the routing and the routing tables [14]. By this, different transient forwarding black holes and/or loops may be created, affecting so the delay performance of

the Internet. Routing protocols like open shortest path first (OSPF) and border gateway protocol (BGP) are especially sensitive to this kind of impairments. Intensive research activity has been started to minimize the negative effects of routing flaps. Some of the best ways to do that seem to be by using increased computing power in routers as well as by using of specific dampening strategies to improve the overall stability of the Internet routing tables and to off-load the CPUs of core routers [14].

3. SERVICE LEVEL AGREEMENT

An SLA is a formal definition of the relationship that may exist between a supplier of services and customers. SLA addresses issues like the nature of service provided, reliability, responsiveness, the process for reporting problems, time frame for response to reported problems and problem resolution, methodologies for auditing service levels, penalty and escape clauses [12]. An internet service provider (ISP) may provide specific SLA to customers who may use the ISP network in different ways, which can be generally described as a combination of three basic modes. These are to access the public Internet, to interconnect two or more sites, and to access proprietary, industry specific networks, e.g., enterprise networks. The terms of SLA that govern each of the access profiles are different. For example, when the objective is to connect two sites of a given customer, the assurances about the performance level will likely focus on the path between the pair of access routers. On the other hand, when the goal is to access the Internet, the ISP may not be in a position to control the performance obtainable but may certify that the performance will be more or less decided by the external Internet cloud and that the ISP's access network will not be the bottleneck.

In the context of computer networks, service specification for SLA can be done at different protocol layers. At the application layer, the focus is on application sessions. Primitives like throughput, object access delay, and transaction update turnaround time are relevant here. Further, the overall performance aspects obtained at the application layer is a combination of the stochastic nature of payload contents sizes, the characteristics of application layer protocol, and of the transport and lower layer protocols. Application layer protocol usually works as a feedback loop on the end-to-end path and thus influences the performance significantly.

At the transport layer, primitives like connection throughput and goodput are considered for SLA. On the contrary, at the link layer, the focus is on packets. Major parameters considered here are bandwidth allocation, rate and burst control issues and port buffer sizing. Depending on particular environments, a combination of layer specific SLAs may be chosen.

Generally, there are two aspects related to SLA of networked systems: availability and responsiveness [15]. They are applicable at the application level and

the network layer. Host or router uptime, link outage frequency, and error rates, they all fall in the category of system availability. Most of these parameters can be monitored and detected using diverse network management tools (e.g., SNMP). Further, some of the major parameters that can be grouped under the category of responsiveness are: one way end-to-end delay at the link layer; application level turnaround time; transmission control protocol (TCP) connection setup latency; TCP retransmissions; packet loss rates; and available bandwidth. Some of these metrics can be collected from regular simple network management protocol (SNMP) or remote monitoring (RMON) type statistics databases whereas others (e.g., TCP and application layer metrics) can only be audited via dedicated monitors.

The perception of the end-user when accessing Internet services is mainly concerned with reliability and responsiveness. The main concern in reliability aspects of Internet is with irregularities in domain name resolution (DNS). The associated criterion for WWW is the probability of lost transaction. On the contrary, the main criterion in responsiveness is the WWW page access turnaround time. Several statistics like mean, variance, maximum, minimum and peakedness can be used to characterize the turnaround time of a Web-based transaction. The most important parameters that influence the WWW page access turnaround time are the protocol type (HTTP/1.0 [16] or HTTP/1.1 [17]), the size of the downloaded Web page, the number and the size of embedded objects downloaded together with the main object, the workload of the server, available throughput rates, long range dependence (LRD) properties of the traffic, utilization profiles, TCP and link-layer controls as well as sizing of buffers [18].

Finally, there is a third part that participates in Web-based transactions, namely content providers. The main performance concerns for content providers are regarding the number of visits, workloads and bottlenecks for the client side (with or without cache proxy server) and for the server side (single or mirrored web servers). Questions related to workload forecasting and capacity planning are relevant for content providers as well.

4. HTTP TRAFFIC MEASUREMENTS

HTTP is a stateless protocol [16, 17]. A stochastic marked point process is used to model the HTTP session and the associated timings (Fig. 1). A HTTP session is said to be the transaction for downloading a single Web page with the associated main and secondary (embedded) objects. The session starts when the user types Web link of the page in the browser and clicks “enter.” At this epoch, the client side establishes a TCP connection with the Web server and, after this, the transfer of HTTP messages between client and server starts. The Web server responds to the client with the requested page and all embedded objects in it. The HTTP session is said to be terminated when the last object is transferred

Fig. 1. Structure of HTTP session.

from the Web server to the client. Once the session terminates, the user screen is complete and shows the Web page and embedded objects. The user takes then time to read the page and clicks on some link in the same page or requests for a new page by typing a Uniform Resource Identifier (URI). This time is the user-think time. It depends upon human behavior and it is defined as inter-session time (passive off time). Furthermore, there is a time gap between successive fetches of embedded objects during the same HTTP session. This is defined as inter-arrival time (active off time). Thus, the timing parameters associated with the HTTP session are defined as follows (Fig. 1):

1. *On time*: time duration for transferring the main object and all embedded objects in it from the Web server to the client.
2. *Passive off time*: time duration between the end of one session and the start of the next session.
3. *Active off time*: time duration between successive fetches of embedded objects within the same *On time*.

HTTP applications have therefore an ON–OFF behavior and, to model this, it is required to gather protocol message elements along with the associated timing parameters. Towards this goal a passive measurement system has been developed to collect HTTP traffic on the BIT student access network. We have followed principles presented in [19] and adopted an “enterprise” point of view in doing

our measurements. Packet traces have been collected by using *Tcpdump* [20]. The traces have been mapped to TCP flows by using *Tcptrace* [21].

To handle the large amount of data collected, it is necessary to make brief summaries of the HTTP flows. Log files of each HTTP session have been created, which contain HTTP flows. HTTP is a stateless protocol and as such it does not maintain information about the session progress. This is because each request for a new web page is processed without any knowledge of previous requested pages. To measure various HTTP session parameter functionality and the associated message timings, the *Tcptrace* is augmented.

Many Internet applications involve protocol messages that are carried as a single packet while the payload is often carried in the form of multiple packets. The *Tcptrace* software has been augmented to facilitate the collection of application layer protocol message timestamps; more specifically to get HTTP session arrival timings and the duration of each HTTP session. Furthermore, it is a challenge to distinguish between different HTTP sessions by just looking at the TCP connections. The silence period between consecutive requests can be used however to distinguish between different sessions. Assumption is made that if the silence period between consecutive HTTP requests exceeds 5 s, a new session is assumed to be started. The logic behind discrimination of consecutive HTTP sessions is depicted in the Fig. 2. C code patch was written in `http_done` of *Tcptrace*, which uses logic as shown in Fig. 2. This generates summary of HTTP flows per session and stores it in separate log files. Figure 3 shows an example of sample log file generated by the augmented *Tcptrace* tool.

The first line represents summary for the request message and the second line represents the corresponding response. The first field represents the time when request was sent from client and the relative time when response to this specific request was received. The first field in the first line shows therefore zero as a start of request and the first field in the second line shows elapsed time, i.e., 0.13511 s. The second field shows the connection identifier `nc2nd`, which means connection from client `nc` to server `nd`. Furthermore, `nd2nc` means connection from server `nd` to client `nc`. The third field is the message identifier where `msg.rqst` represents request message and `msg.repl` represents reply message. The fourth field represents the size of the header block for either request or response message. The fifth field represents the content size in bytes. As request message has no content, fifth field for the request message shows zero. The sixth field shows the packet number. This is the number given by the *Tcpdump* while creating the dump file of the network traffic. The seventh field shows the HTTP version of the client for request message and of the server for the response message. The eighth field shows the method for the request message and the response code for the response message. The ninth field for the request message shows the host address where client sent request and for response message it shows the type of object. Finally, the tenth field in case of request message shows the referrer page for the given request. In this way

Fig. 2. Flowchart for the generation of HTTP session log files.

augmented *Tcptrace* generates summary of output that can be used for extracting the distributional properties of application layer parameters.

5. HTTP TRAFFIC ANALYSIS

The augmented *Tcptrace* produces summary of HTTP sessions in form of log files. By analyzing the log files, various statistical properties of the HTTP traffic can be observed. The largest part of HTTP traffic is generated from the server

```

0.000000 nc2nd msg.rqst 573 0 6587 HTTP/1.1 GET 164.109.168.120:80 Unknown
0.135112 nd2nc msg.repl 18457 824 6622 HTTP/1.1 200 text/html
0.725952 ne2nf msg.rqst 465 0 6618 HTTP/1.1 GET 164.109.168.120:80 Unknown
0.848140 nf2ne msg.repl 348 43 6620 HTTP/1.1 200 image/gif
1.084819 ng2nh msg.rqst 471 0 6627 HTTP/1.1 GET 212.72.38.145:80 Unknown
1.154107 nh2ng msg.repl 373 43 6631 HTTP/1.1 200 image/gif
  
```

Fig. 3. Sample http.Flowstat output.

toward the client. The properties of HTTP traffic can be categorized in two broad terms as follows:

- (1) *Content Properties*: the Web page often contains embedded objects in terms of images, sound and video rather than a plain text. The content property is a notion for the servers that hold pages and involves the number of Web pages it holds.
- (2) *Structural Properties*: the structure of a Web page varies from server to server. News Web server generally has pages filled with lots of embedded items like images and video clips having small sizes. On the other hand, educational institutes usually have Web pages with less embedded objects of large sizes like reports, presentations, etc. The main parameters for the collected HTTP traffic have been analyzed as follows.

5.1. Inter-Session Arrival Timings (Passive Off time)

This time represents the user behavior and it is known as user think time. The measurements done at the student access network of BIT, Karlskrona, Sweden have shown that the arrivals for requests for WWW sessions can be well modeled by a renewal process, and for the most of cases this is a Poisson process. It has been observed that the inter-session arrival timings can be modeled in this case

Fig. 4. Histogram of inter-session arrival timings (passive off time).

by an Exponential process with a mean of 18.5490 s and a median of 12.4640 s (Fig. 4).

5.2. Inter-Arrival Timings (Active Off time)

This time represents the duration between successive fetches of embedded objects within one HTTP session. This parameter depends on the browser application running at the client side. Normally, HTTP requests are issued sequentially, with the next request being issued only after the response to the current request is completely received. However, in the case of HTTP/1.1 with pipelining, the client may open several parallel TCP connections with the server, and in such a case the inter-arrival timing is very small. It has observed that HTTP/1.0 without persistent mode may also open parallel TCP connections. Our measurements have shown that these timings mostly lie between 10 and 600 ms as shown in Fig. 5.

5.3. Request Message Size

Client sends request to the server for fetching web page in the form of request message. The server responds to request in form of response message. It is observed

Fig. 5. Plot of inter-arrival timings (active off time).

Fig. 6. Plot of request message size.

that most of the time the request message is of approximately 450 bytes. Furthermore, it has been also observed that sometimes the request message can be longer when the user wants to send authentication information or online forms. In such a case the request message size often goes beyond 450 bytes as shown in Fig. 6.

5.4. Response Code

This parameter gives information about the success and the failure rate for accessing the server. Our measurements have shown that the success rate is approximately 88%. Figure 7 shows the histogram of response code. It is observed that response code 200, which is considered as success, has the maximum frequency.

5.5. Number of Transactions

Often Web pages contain embedded objects in the form of image, sound or video files. The number of embedded objects is termed as number of transactions. It is possible to count the number of transactions within a HTTP session by counting the number of requests the client sends towards the server. This parameter provides information about how heavy the Web page is and

Fig. 7. Histogram of response code.

also interprets the session duration. It has been observed that the Web pages retrieved at the student access network of BIT are more likely to be filled with embedded objects and the histogram for number of transaction gives a mean of 23.9560 and a median of 6 as shown in the Fig. 8. It has been also observed that the number of transactions can be modeled by a negative binomial distribution [18].

6. CONCLUSIONS

A measurement and analysis study of HTTP traffic collected from the student access network of BIT, Karlskrona, Sweden has been reported. *Tcpdump* has been used to collect traffic and the dump file has been fed to an augmented *Tcptrace* to generate various log-files per HTTP session. The study has revealed user, client, and server behavior. The inter-session gap between consecutive HTTP sessions is user dependent. It has been observed that under normal conditions this has a mean value of 18 s. The inter-arrival gap, which often depends upon client's browser application and the pipelining capabilities, has been found to be in the range of 10–600 ms. The request message size has been observed to vary from one request to another and it has been observed an average request message size of about

Fig. 8. Histogram of number of transactions.

450 bytes. When the user sends some information with request message, the size of the message goes beyond 450 bytes.

The properties of the HTTP server are obtained from the response code, which gives statistics about the success or the failure rate for the client's request. The success rate has been observed to be approximately 88% and this can differ depending upon the specific server to which the client send requests. The heaviness of the Web page residing on server can be judged by analyzing the number of transactions within a HTTP session. It has been observed an average of 24 embedded items in a Web page. This parameter depends upon the structure of Web page and the type of organization hosting the specific Web page.

ACKNOWLEDGMENT

The authors gratefully acknowledge Dragos Ilie and David Erman at BIT, whose comments were helpful for handling various practical issues.

REFERENCES

1. Lenny Zeltser, The World Wide Web: Origins And Beyond, April 1995, <http://www.zeltser.com/WWW>.

2. M. Arlitt and C. Williamson, Internet Web Servers: Workload Characterization and Performance Implications, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 5, October 1997.
3. B. A. Mah, *An Empirical Model of HTTP Network Traffic*, IEEE INFOCOM, Kobe, Japan, April 1997.
4. K. Heideman, K. Obraczka, and J. Touch, Modeling the Performance of HTTP Over Several Transport Protocols, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 5, October 1997.
5. I. Cooper and J. Dilley, Known HTTP Proxy/Caching Problems, RFC 3143, June 2001.
6. P. Barford and M. Crovella, Generating Representative Web Workloads for Network and Server Performance Evaluation, ACM SIGMETRICS, 1998.
7. M. Z. Hasan and S. Lu (eds.), Special Issue on Internet Traffic Engineering and Management, *Journal of Network and Systems Management*, Vol. 10, No. 3, September 2002.
8. G. Lin and C. Shen, Special Issue on Management of Converged Networks, *Journal of Network and Systems Management*, Vol. 10, No. 1, March 2002.
9. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, An Architecture for Differentiated Services, RFC 2475, December 1998.
10. G. Armitage, *Quality of Service in IP Networks*, MacMillan Technical Publishing, USA, 2000.
11. S. Casner, V. Jacobson, Compressing IP/UDP/RTP Headers for Low-Speed Serial Links, RFC 2508, February 1999.
12. A. K. Jena, A. Popescu, and A. A. Nilsson, Resource Engineering for Internet Applications, International Conference on Advances in Infrastructure for Electronic Business, Science, Education, Medicine, and Mobile Technologies in the Internet, SSGRR 2003w, L'Aquila, Italy, January 2003.
13. W. Willinger, V. Paxson, and M. S. Taqqu, *Self-Similarity and Heavy-Tails: Structural Modeling of Network Traffic*, Birkhäuser, Boston, 1998.
14. S. R. Sangli, Y. Rekhter, R. Fernando, J. G. Scudder, and E. Chen, Graceful Restart Mechanism for BGP, Internet Draft, April 2002.
15. D. Verma, *Supporting Service Level Agreements on IP Networks*, Macmillan Technical Publishing, USA, 1999.
16. T. Berners-Lee, R. Fielding, and H. Frystyk, Hypertext Transfer Protocol—HTTP/1.0, RFC 1945, May 1996.
17. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, Hypertext Transfer Protocol—HTTP/1.1, RFC 2616, June 1999.
18. A. K. Jena, A. Popescu, and A. A. Nilsson, Modeling and Evaluation of Internet Applications, ITC18, Berlin, Germany, August/September 2003.
19. N. Brownlee, Network Management and Realtime Traffic Flow Measurement, *Journal of Network and Systems Management*, Vol. 6, No. 2, June 1998.
20. *TCPDUMP* public repository, <http://www.tcpdump.org>.
21. Shawn Ostermann, *TCPTRACE*, Tool for analysis of TCP dump files, <http://www.tcptrace.org>.

Yogesh Bhole received Master of Science in Electrical Engineering with emphasis on Telecommunications from Blekinge Institute of Technology, Karlskrona, Blekinge, Sweden in August 2004 and Bachelor of Electronics Engineering from VJTI, University of Mumbai, India in June 2003. He is enrolled in Master's program in Applied Electronics with a major in Wireless Communication at Umeå University for the academic year 2004–2005. His research interests are Internet traffic measurement and analysis, Mathematical modeling of network traffic, Ultra Wide Band Communication.

Adrian Popescu received two Ph.D. degrees in electrical engineering, one from the Polytechnical Institute of Bucharest, Romania, in 1985 and another from the Royal Institute of Technology, Stockholm, Sweden in 1994. He has also the Docent degree in computer sciences from the Royal Institute of Technology, Stockholm, Sweden (2002). He is an Associate Professor in

the Department of Telecommunication Systems, Blekinge Institute of Technology, Karlskrona, Sweden. His main research interests include Internet, communication architectures and protocols, traffic measurements, analysis and modeling, traffic self-similarity, as well as simulation methodologies. He is a member of the IEEE, IEEE CS, ACM, and ACM SIGCOMM.