Workload Characterization of Uncacheable HTTP Content

Zhaoming Zhu, Yonggen Mao, and Weisong Shi

Department of Computer Science
Wayne State University, Detroit, MI 48202, USA
{zhaoming,ygmao,weisong}@wayne.edu

1 Introduction

The rapid growth of uncacheable content over the HTTP protocol [1, 2] necessitates the further investigation and exploitation of its properties. In this paper, we intend to answer these following questions: Among the huge HTTP content delivered on the Internet, which parts are cacheable and which parts are uncacheable? What are their characteristics, especially for uncacheable content? Is there any difference among different uncacheable HTTP content? Is there any cacheable possibility for these conventional uncacheable content? How about the cacheability of personalized content? Is there some relationship between uncacheable content and HTTP persistent connection?

To answer these questions, we sniffed and analyzed all inbound and outbound HTTP traffic on all possible TCP ports at a medium-size education institution. By analyzing an one-day trace, we observed the following: (1) uncacheable data have dominated today's HTTP traffic and multimedia type content transferred by P2P applications (35.5%) and graphic type (jpeg and gif) content (17.3%) are top two in the uncacheable HTTP objects; (2) compared with cacheable content, uncacheable content consumes more server-processing time, but due to network latency, the client-perceived response time tends to be close to that of cacheable content; (3) on average, uncacheable content has an larger object size than that of cacheable objects (13 K bytes vs. 7K bytes); (4) clients accessing personalized content and servers providing personalized content are more concentrated than general clients and server groups, while the total online personalized content occupies only a smaller percentage (less than 10%), far below than previous observations from [2]; (5) a considerable (50%) portion of HTTP objects have their TTL values equal to zero. Among these objects, we observed that the URL aliasing contributes 20% of total requests; (6) P2P traffic is increasingly scattered among multiple ports (only 13% on default ports for KaZaA traffic), which is a big challenge for deployment of P2P traffic caching. Several implications could be derived based on above observations: (1) a considerable portion of uncacheable HTTP content is cacheable; (2) domination of network latency factor motivates the moving of functionality for uncacheable HTTP content generation to the edge of networks; (3) prefetching for personalized Web content is promising because of the concentrated popularity of clients and servers; (4) convinced by the observed P2P request popularity, we believe that the content-based caching is significant to the ever-increasing P2P traffic; (5) exploiting URL-alias is a promising direction to improve cacheability of uncacheable content.

2 Analysis Results

2.1 Trace Collection

We collected one-day period (12:00 pm, Mar 18 -12:00pm Mar 19, 2003) HTTP traffic, rebuilt and investigated the contained content. To capture all possible HTTP traffic, TCP packets on all ports were sniffed. We have developed WebTACT, an **W**eb **T**raffic **A**nalysis and **C**haracterize **T**ool, to extract the complete HTTP information, including both header and content. Analysis results and observations are depicted in the following. Due to space limitation, we present high level results only, more detailed data and analysis can be found in the technical report version of this paper [3].

From the viewpoint of proxy caching, generally HTTP object could be broadly categorized as uncacheable content or cacheable content. The cacheable HTTP objects refer to those infrequently changed HTTP objects (also known as static HTTP content), and the uncacheable HTTP content could be further classified into uncacheable subtypes one to seven respectively: `NonGet`, `DynGen`, `Pragma`, `CacheCtl`, `Personalized`, `AbnormalStatus` and `ZeroTTL`, based on the HTTP protocol specification [4].

2.2 High Level Characteristics

Table 1 lists the high level statistics for both cacheable and uncacheable content. For each content type, we detail them in different traffic directions. The inbound traffic means that the response objects are targeted to clients inside the campus, while the outbound traffic means that the response objects are targeted to clients outside the campus. The total distinct client number inside the campus is 9,053, and that outside campus is 93,250. The total server (host providing HTTP content) number inside the campus is 1,930, and that outside the campus is 114,416.

From Table 1, we can see that, the captured-reconstructed gross HTTP traffic (include HTTP headers and bodies) is around 117.5 GB. For the total objects size (or the total size of transferred HTTP response objects),[1] the uncacheable content outnumbers the cacheable content (72 GB vs. 2.7 GB). The servers that providing uncacheable content outnumber those providing cacheable content (116,149 vs. 7,518), while the clients accessing dynamic content also largely outnumber the clients accessing cacheable content(101,971 vs. 14,674). These data exemplify that the uncacheable content dominates today's HTTP traffic and the necessity of efficient delivery of uncacheable content. The majority of HTTP uncacheable traffic is multimedia audio/video type. This is because that: (1) the huge volume of P2P (KaZaA) application traffic focuses mainly on multimedia file exchange; (2) 99.6% KaZaA HTTP objects are categorized into uncacheable type by our analyzer. Comparing our data with previous results in [2], we observe an increase in the uncacheable request/response for image (`gif` and `jpeg`) content type, and a decrease in text (`html` and `plain`) content type. The possible reason is the widely acceptance of cache busting technologies [4]. The multimedia type objects (`video/x-msvideo`, `video/mpeg` and `audio/mpeg`), which contribute to a large percentage of total bytes and a small percentage of total number of responses, implies a larger average size of these kinds of objects.

[1] Hereafter, all the traffic mentioned in this paper are referring to total objects size.

Type	Cacheable		Uncacheable	
HTTP Traffic Direction	Inbound	Outbound	Inbound	Outbound
# of Servers	7,345	173	114,221	1928
# of Clients	7,007	7,667	9,050	92,921
Total Gross Traffic(MB)	2,177	875	66,278	51,017
Total Object Size (MB)	1,917	825	42,209	31,619
# of Requests	309,616	73,662	6,688,378	2,181,252

Table 1. High-level statistics of HTTP traffic.

2.3 Detailed Characteristics of Uncacheable HTTP Content

Uncacheable Content Breakdown Table 2 lists the absolute numbers for each of the seven uncacheable subtypes, by their total object size and request/response number. The "mixed" type means the uncacheable subtype is a combination of this subtype and at least one other uncacheable subtype, while the "pure" type means the request belongs to this subtype only. We find that the `personalized` objects (subtype 5) occupy less than ten percent of all uncacheable content in terms of both bytes and number of requests, not as large as previous observation. We do not know the exact reason for this low percentage of personalized HTTP objects. A distinguished portion, `ZeroTTL` (subtype 7), implies a promising probability of caching performance improvement that we will give more detail analysis later.

Subtype	ZeroTTL	AbnormalStatus	Personalized	CacheCtl	Pragma	DynGen	NonGet
pure requests	4,413,886	2,067,218	71,634	104,186	80,947	1,274,334	69,767
pure size(MB)	58,182	1,167	541	974	2926	6,493	276
mixed requests	0	574,645	92,989	402,960	232,048	476,420	80,710
mixed size(MB)	0	348	398	2,360	72	2,819	352

Table 2. Detail breakdown for all seven uncacheable subtypes.

Response Time and Breakdown for Uncacheable Content For further analysis, we first want to know, whether the cacheability of objects affects their response time, on both server side (processing time) and client side (latency). Because of the sniffing point location of our study, we could assume that for the inbound traffic, the response time is close to client-perceived latency, and for the outbound traffic, the response time is close to server-processing time. For the inbound HTTP traffic, the difference between the response time of uncacheable and cacheable objects is not so much. This implies that the time difference caused by dynamic/static content generation has been blurred by the network latency on their route. For the outbound HTTP traffic, there is a difference between the response time of uncacheable and cacheable content, this is probably caused by the time necessary to dynamically generate the uncacheable content. The response times for different dynamic types do not show much difference, especially for inbound traffic. These phenomena show the need to migrating the dynamic generation functions to the network edge.

Object Size Distribution The distribution of object size is also an interesting topic, especially when HTTP objects are classified into two major classes: cacheable and uncacheable. Intuitively we believe that, on average, uncacheable object size is smaller than cacheable size, but our analysis gives contrary result. 90% of cacheable objects smaller than 14,860 bytes, while same percentage uncacheable objects are smaller than 18,694 bytes. The average size is 7K bytes (cacheable) vs. 13K bytes (uncacheable).

Amazingly, the largest HTTP object size we observed is 252 M bytes for uncacheable object and 12M bytes for cacheable objects. These numbers are much smaller than those appear in [5]. The possible reasons are: (1) our data collecting period is relatively short (24 hours vs. 9 days data collecting period [5]); (2) the object size is calculated based on the bytes on the wire, instead of the HTTP headers. As more and more applications (e.g., KaZaA) adopt parallel downloading or other segment-based content delivery techniques, supported by the HTTP protocol, we believe the size of individual HTTP objects will not be larger any more. So the real reconstructed (fragmented) objects reflecting only a fraction of total size is a reasonable explanation.

Is Uncacheable Content Really Uncacheable? Although the object composition technique, such as ESI , has been proposed, in this paper we are looking for URL-alias derived cacheable possibility. Totally, there are 4,413,886 objects belonging to ZeroTTL subtype. Among these we observe that many different URLs share the identical content digest. This is caused by the phenomenon called "URL-alias" [6]. Our analysis show that the number of requests targeting the top 1000 (less than 0.1% of total distinct digest value) rank digest value count for 18% of total number of requests. This observation reveals an opportunity for the future Web cache improvement if certain protocol could be designed to deal with ZeroTTL objects based on their digest value, rather than on their URLs only.

Client/Server Popularity We assume that the personalized Web content would be more client/server-specific than general content, due to its "personalized" property and the analysis results do verify our assumption. Top 1% of clients that accessing personalized content bring about 20% of the total personalized content requests. However, unlike previous observations [2], we find that clients interested in personalized content only occupy 2% of the total client population. Some clients are much more likely to access personalized Web content. These clients are some public-access computers, located at public area like student dormitories, for students check updated personalized information like email or personal account on e-commerce Web sites. We also find that personalized content is provided by 1% of the total servers, and servers providing personalized content are also more concentrated than server providing general content. Top 1% of servers that provide personalized content handle 85% of the requests for personalized content requests. There are "hot" personalized Web servers and the top 30 of the servers contribute 95% of the total requests among the top 100 servers providing personalized content.

Object Popularity Due to the personalized property, personalized HTTP objects might not be more concentrated than general objects and this is proved by our analysis. All these observations strongly suggest that personalized prefetching could be used to reduce the client perceived latency and network bandwidth equipment.

Persistent Connection vs. Uncacheable HTTP Objects In our reconstructed HTTP data, there are totally 669,958 persistent connection sessions, consists of 3,411,741 HTTP request/response pairs. On average, a persistent session consists of 5.09 pairs. We have supposed that the uncacheable content would have some distribution patterns among multiple pairs within one persistent connection. One reasonable assumption is that, for a persistent HTTP connection, maybe the first object is an uncacheable dynamically generated page template, followed with embedded cacheable objects like graphics. But our analysis results deny this distribution pattern proposition and conclude that most of the uncacheable content does not appear in persistent connections.

Peer-to-Peer Traffic Analysis We reconstructed all http-based P2P traffic by capturing all TCP traffic, instead of sniffing only some specific default ports (e.g., 1214 for KaZaA, 6346 and 6347 for Gnutella, used by previous work [5]). Generally, as observed earlier in this Section, P2P traffic contributes to a large portion of total HTTP traffic. For Gnutella type P2P applications, we aggregate several found Gnutella client applications (e.g., `LimeWire`, `BearShare`, `Shareaza` etc.) into a whole Gnutella division. For KaZaA type data, only the traffic from KaZaA client is calculated. The total HTTP object size transferred by Gnutella applications is 1,110,383,667 bytes, while that by KaZaA application is 26,659,686,069 bytes. The total object size transferred by P2P applications occupies 33.8% of the total observed HTTP object size while the corresponding percentage is over 75% in [5]'s work. With the evolution of P2P applications, P2P traffic ports are more distributed than before. For example, only 13% of KaZaA traffic is through its default port 1214. This phenomenon strongly implies the emergence of new mechanisms dealing with P2P traffic spreading on different TCP ports.

3 Summary

Implied by characteristics analysis of uncacheable HTTP traffic, we propose four promising directions to improve caching and content delivery of uncacheable HTTP content: first, pushing the functionality of uncacheable content generation to the network edges; second, applying the access pattern feature to prefetching schemes; third, implementing an efficient content-based P2P traffic caching; Finally combining content-based approach into current cache to exploit the prevailing URL-alias phenomenon.

References

1. Shi, W., Collins, E., Karamcheti, V.: Modeling object characteristics of dynamic web content. Journal of Parallel and Distributed Computing) **63** (2003) 963–980
2. Wolman, A., Voelker, G.M., Sharma, N., Cardwell, N., Brown, M., Landray, T., Pinnel, D., Karlin, A., Levy, H.M.: Organization-based analysis of web-object sharing and caching. In: Proc. of the 2nd USENIX Symposium on Internet Technologies and Systems (USITS'99). (1999)
3. Zhu, Z., Mao, Y., Shi, W.: Workload characterization of uncacheable http traffic. Technical Report MIST-TR-03-003, Computer Science Department, Wayne State University (2003)
4. Krishnamurthy, B., Rexford, J.: Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching and Traffic Measurement. Addison-Wesley, Inc (2001)
5. Saroiu, S., Gummadi, K.P., Dunn, R.J., Gribble, S.D., Levy, H.M.: An analysis of internet content delivery systems. In: Proc. of OSDI'02
6. Kelly, T., Mogul, J.: Aliasing on the world wide web: Prevalence and performance implications. In: Proc. of the 11th International World Wide Web Conference (2002). (2002)