INTRODUCING TRANSPARENT WEB CACHING IN A LOCAL AREA NETWORK

Marina Buzzi, Laura Abba - CNR, Istituto per le Applicazioni Telematiche
Damir Pobric - Consorzio Pisa Ricerche
Massimo Ianigro - CNR, Area della Ricerca di Bari

The term "transparent web caching" refers to cache technology in which web traffic is automatically intercepted and redirected toward one or more cache servers. The redirection of web data can be accomplished using L4 switches or routers. Being completely transparent to the user (no browser configuration is required) the service can be easily implemented and turns out to be scalable and fail-safe. This work presents the results of our experimental use of transparent caching technology in a simple network environment. We focus on the impact this technique could have on network performance, with all its benefits and problems, as well as its effects on end users. The analysis is based on data gathered in an operative network during a two-month period.

1. INTRODUCTION

The ever-increasing volume of web traffic is placing tremendous demands on the Internet; and the web services have become a primary consumer of network bandwidth. In order to reduce network and server load and to improve web document (Internet object) access latency two solutions have been widely deployed:

- web caching, extensively used in Europe and
- web replication, widespread throughout the United States.

Both technologies aim at reducing network congestion phenomena and improving the Quality of Service (QoS) offered to the user, while employing different approaches.

Web caching technology stores temporary copies of web objects on an intermediate server closer to the user than the original web server. If more than one user wishes to access the same objects the cache server can reduce the time elapsed between the user request and the object's availability. Cache servers could co-operate in either hierarchical or flat architectures, sharing a larger set of Internet objects. By shortening the network path to Internet objects, web caching reduces network traffic and optimizes the bandwidth usage. However, the use of temporary copies could generate inconsistency between the original object and the stored copy.

Web replication is based on the redundancy concept: geographically widespread web servers all offer the same services. A user is bound to the "most convenient" replica server, depending on the load distribution policy used. Web replication does not reduce web traffic, but divides it up over different paths and servers.

Due to increased interest in these technologies, last year the Internet Engineering Task Force set up the caching and web replication (wrec) working group.

In this paper we focus on web caching technologies and on transparent web caching in particular.

2. TRANSPARENT WEB CACHING

Transparent caching is a variant of web caching where web traffic is automatically intercepted and redirected toward one or more web cache servers, using L4 (or L5) switches or routers (Figure 1). The switch transparently intercepts web traffic originated by the clients and applying a load balancing policy redirects http requests toward one of the available cache servers. If a server become unreachable, the switch rearranges the http traffic over the remaining active cache servers. The traffic not addressed to port 80 is sent to the router. Switches can be used to keep HTTP traffic carrying unknown methods (i.e. http protocol extension) out of the cache. In order to divert unknown HTTP methods, or to apply more sophisticated routing techniques, the switches must be capable of snooping the incoming data streams at the application level (Layer 5-7).

Routers can accomplish data redirection in two ways: statically (policy based) or dynamically, by the use of the Web Cache Coordination Protocol (WCCP)

described later.
Transparent caching offers multiple advantages:

- It requires no client configuration, thus dramatically reducing the administrative tasks[1].
- Traffic is automatically rerouted; users cannot bypass the cache service.
- The service is fail-safe, since the switch (or router) can bypass the cache server if it should happen to be down.
- The architecture is scalable.
- Load balancing between cache servers can be applied;

It can co-operate smoothly with other servers within the cache hierarchy.

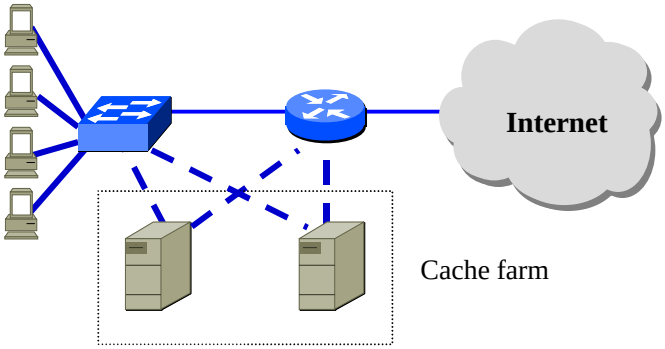

Fig. 1 - Transparent caching (logical scheme)

However, transparent caching has a few disadvantages:

- It does not cache ftp traffic (usually large files).
- Transparent caching can break client cache directives. The problem is discussed in [DILL00]. The lack of cache directives in an http request can generate data inconsistency. If the client is unaware of the connection to a cache server it may not include critical cache-control directives such as "no-cache" or "must-revalidate" in the http request. The omission of these cache-control directives can cause the client to receive data that is not synchronized with the origin object, even if the user had requested data revalidation.

1 The automatic proxy configuration is a simple, flexible, scalable and fail-safe technique based on the use of a javascript (.pac file) to define the web client policy for access internet objects [PAC96]. The web client downloads the pac file from a web server. WPAD (Web Proxy Auto Discovery) [GAUT99] permits automatic discovery of the URL where the javascript is stored, thus further simplifying configuration tasks, without however eliminating them. Unfortunately the latter method is currently available only for Microsoft Internet Explorer.

2.1. The WCCP protocol

The WCCP (Web Cache Coordination Protocol) is a protocol designed by CISCO. The protocol is used to associate a router with a cache farm to redirect web traffic in a transparent way. In order to balance the load, a designated web-cache (the one with the low IP address) decides which policy will be deployed by router to redistribute traffic.

The WCCP version 1 was published as an internet draft [CIES99] and some proxy caching products (including squid) already implement the support of the protocol. This first version suffered from performance problems, due to an increasing CPU usage on the router. These problems were resolved in the next version.

Version 2 of the protocol offers additional features [CEUG]:

- Multiple router support;
- Improved security;
- Faster throughput;
- Permits the redirection of multiple TCP port-destined traffic;
- Load-distributing applications capability;
- Client IP addressing transparency.

It is supported by CISCO IOS 12.0(3)T and 12.0(5)T. The protocol specification is now available as an Internet draft [CIES00], but squid does not support it yet.

3. STUDY

This paper describes the first part of a work in progress concerning transparent caching. We started with the simplest case, using a single stand-alone cache. This will eventually be joined to the CNR cache hierarchy. In order to evaluate both efficacy and any problems that may arise when this technique is introduced in an operative environment, we configured and tested transparent web caching on a LAN of the Institute IAT, CNR Research Area in Pisa.

First, we will briefly describe the environment where measurements were recorded.

3.1. The network

The Network of the CNR Research Area in Pisa is a nice example of building a campus network upon flexible ATM technology. The core of this rather complex network infrastructure is a backbone composed of 6 ATM switches, covering all three of the

Area's buildings and interconnected by 155Mbps redundant fiber links. The LANs of each of 15 Institutes are built on a variable number of 10/100 Ethernet switches with ATM up-link(s) to the backbone. Interoperability between two environments is obtained through LAN Emulation. At the top of the hierarchy is a unique router. This is connected to the network through a single ATM interface on one side and provides the Area's community with Internet connection (4Mbps link to network GARR) on the other. The logical representation of the entire network is illustrated in Figure 2.

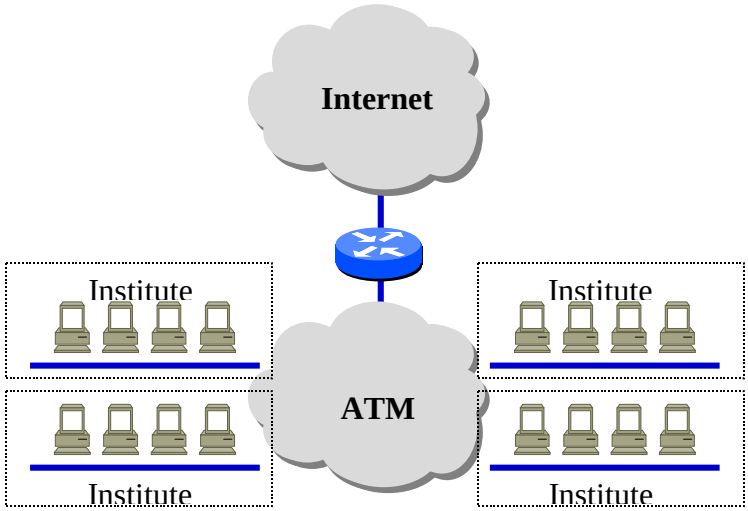

Fig. 2 - The network of the CNR Research Area of Pisa

3.2. The Cache Server

We initiated our test in a low-cost environment. The cache server runs on a Pentium III-450 Mhz 512kb cache, 512 Mb of RAM memory, U/Dma 14 Gb disk. The OS is Linux Red Hat 6.1, kernel version 2.2.12-20 and the proxy software is Squid, release 2.2-STABLE5. The cache server has only one 10/100 Ethernet port. This port is directly connected to the router by means of a crossed TP cable. The speed of this dedicated LAN is limited to 10Mbps, as the router has only standard Ethernet ports.
The implemented transparent caching topology is depicted in Figure 3.

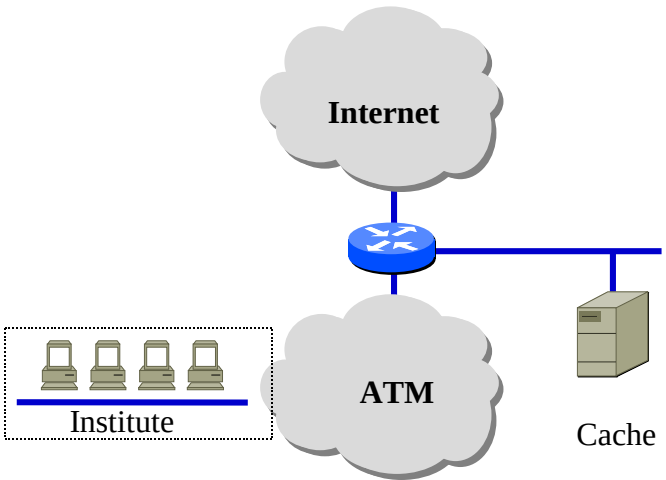

Fig. 3 - The implemented transparent caching topology

SQUID FAQ [WESS00] provides detailed information on configuration of the cache server. We applied the default Squid configuration file modifying only the options affecting the cache memory and disk sizes. We dedicated 8 Gb of disk space and 180MB of memory to Squid.

3.3. The clients

On IAT LAN there are about 100 computers but less than half are used to browse the Internet. The average number of clients accessing the cache server in the monitored period (on working days) was 34.

3.4. Traffic redirection

The HTTP traffic, arriving from clients residing on IAT LAN, is redirected toward the cache server by router. This kind of redirection is called "policy based" (the Cisco term is "policy based routing") and consists of forwarding all the packets destined for port 80 to the cache server. The router does this even if a cache server is down or not functioning properly and, as opposed to that of Layer 4 switches, this kind of redirection does not provide support for fail-over in case a cache server fails[2].

3.5. Objectives

In this paper we attempt to evaluate:

- The web caching service performance
- Pros and cons of transparent caching in one real context
- The impact of this technology on the final user

3.6. Evaluation

We decided to study the effects of transparent caching

2 We can personally attest to this following an unfortunate experience. Due to emergency repairs the (un-interruptable) power supply was suspended and the cache server brutally turned off! A few minutes later, the power supply resumed but the server didn't boot again and remained silent for more than an hour. The reason was very simple: even with the main power unit switch turned on, a PC starts up either by striking the keyboard or by pushing the power button on the front of the chassis. Because of this, the cache server remained inactive until we turned it on manually. Fortunately, all this occurred early in the morning when no users were browsing the network yet.

from multiple points of view embracing performance, saved bandwidth, quality of service perceived by the user, service acceptance by user community and the national privacy law[3]. We started this first phase of testing with a low-cost solution. The analysis of data collected will then be used to define the system requirements for the specific operative environment. The cost/benefits analysis could prevent investment in unnecessary network equipment.

3.6.1. Data sources

The kind of analysis we performed required the collection of data from both cache server and router. These Multiple data sources allows us to:

- Cull all the necessary data;
- Validate data consistency of the same variable gathered from different sources.

Original squid log files

The squid log files include very important information that could be elaborated with various tools. In order to respect the user's privacy we configured squid to anonymize the client IP addresses. We used the Calamaris software [BEER99] to extract the following data:

- The total traffic (requests and kbytes)
- Request hit/miss rate
- Byte hit/miss rate
- Percentage of requests destined to the .it TLD

MRTG

MRTG [Multi Router Traffic Grapher] [OETI00] is a tool for gathering data from network devices via SNMP protocol. The data (MIB variables) are logged into files and represented as graphs embedded into web pages. MRTG typically samples data every five minutes and creates daily, weekly, monthly and yearly graphs; these can be used to monitor the cache server in nearly real-time and to observe long-term trends of cache usage/performance. In order to make SNMP queries to squid it must be first compiled with the “enable-snmp” option. The second step is to configure squid to accept SNMP requests – the default value is to deny SNMP access [GRAB99]

3 The access to data revealing 'sensitive information' about people, such as state of health, religious faith, political opinions and so on is protected in Italy and in the EU by a number of laws (i.e. Legge 31 dicembre 1996 n. 675 and etc.). When deploying a transparent caching system, the administrators should be careful to either clean up or anonymize the log files.

NetFlow

NetFlow is a high-performance IP switching feature of the cisco IOS that also captures a rich set of traffic statistics. The data collected consists of *traffic flows*. Flows are defined as unidirectional sequences of packets between a given source and the destination endpoint (sharing the same protocol and transport-layer information). The captured statistics are than exported from routers and can be used for a wide variety of purposes (network monitoring, analysis and planning, accounting, billing, attack detection, etc.). To collect and analyze exported flows, we used the Cflowd [CFLO]0 package developed by CAIDA. It is apparent that the fine granularity of collected information requires the storage of a great quantity of data. Even though flows are archived in compact format and Area’s Internet line speed is not high the files with daily statistics average 50Mbytes. These files contain the trace of each single byte which has entered or exited the network and permits any kind of precise traffic calculation. However, the utilities for viewing and post processing of the contents of the files have limited filtering and aggregating features and cannot be used for complex data extraction. We used this data to obtain the total amount of data generated/received by the IAT network and the amount of http traffic served by both cache and origin servers.

3.6.2. Data analysis

The percentage of hit rate

The web traffic served by cache varies between 90 and 180 Mb per workday, with lower values during the weekends and holidays. The daily mean value of traffic for the 8-week period (weekends excluded) is about 118 Mb per day. The chart in Figure 4 shows these statistics for April of 2000.
We observed that the (object) hit rate varies from 30% to 50%, with 41,19% as an average value. The bytes hit rate is contained in the range between 12% and 23%, with an average value of 19,23%. This implies that most of the cached objects are small. Duska et al. [DUSK97] observe that (1) smaller objects tend to have a higher hit rate than larger objects and (2) that in cache hierarchy the number of byte hits increases due to the greater number of objects shared.
We expect the cache to yield further benefits after we insert it into the CNR hierarchy and extend the service to the entire Area.

Variation of page cache size

We have already mentioned that we applied the Squid default configuration file where the size of the

cacheable object was 4 Mb (default value).
Based on the results of the first two months, we doubled the maximum_object_size parameter moving from 4MB to 8Mb, in order to increase the byte rate. We sought a balance between saving bandwidth or increasing the speed.
In May and June we observed the median of byte hit rate increased to 20,70 % with an object hit rate of 42,45%. It is also important to notice that during the entire experimental period the cache was populating, thus enlarging the set of cached objects. The populating process took 5 months: from May 2 to August 2 the cache was filled passing from 34% to 90%.

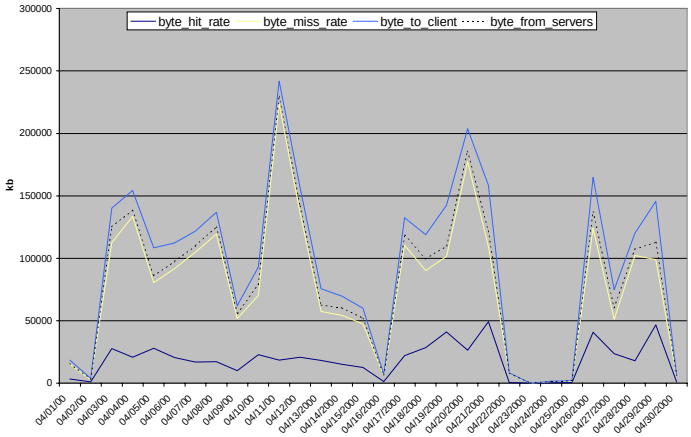

Fig. 4 - Cache statistics, April 2000

The changes in Internet bandwidth utilization

Figure 5 shows the total traffic coming into our network. The web traffic curve (Mbytes_to_clients) is very low with respect to the total traffic entering the network. There is no anomaly in this: the main CNR e-mail server, hosting the e-mail service for several CNR Institutes, resides on the IAT network. In addition, other services such as mail lists, fax gateway and document format conversion are present on this system. This, however, does not affect the benefits of web caching. Although the saving of network bandwidth is significant, this does not lead to a cost reduction in the fixed fee we pay for Internet access.

The .it TLD traffic

The analysis of squid log files helped us to better understand the interests of the user community and the volume of data downloaded from TLDs. The .it TLD is the most frequently visited, followed by the .com. A total of 40 % of bytes requested, corresponding to 54% of objects requested, was addressed to the .it TLD.

The latency introduced by router and cache server

In the previous paragraph we showed that deployment of cache servers had positive effects on bandwidth consumption and that savings were considerable. We also calculated that the average download time of web objects decreased, thus significantly improving the latency. However, in the unfortunate case when the requested web object is not available in the cache and must be retrieved from the origin server, latency increases as the packets regarding that http connection must travel twice over the router, cache-router network segment and cache server itself. We tried to calculate and/or estimate this latency and its effects on perceived quality by end users.

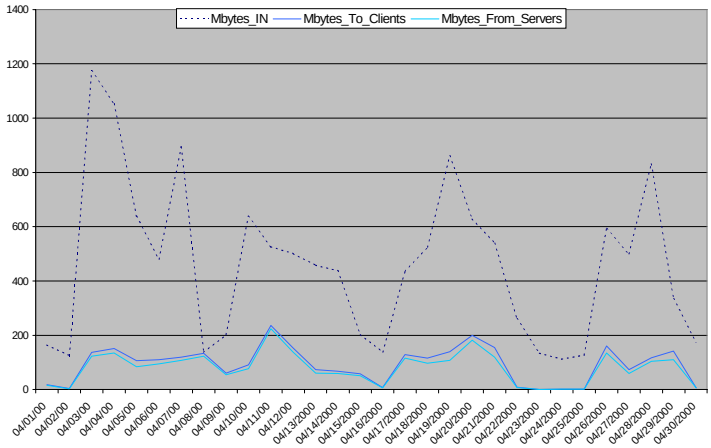

Fig. 5. Total network traffic vs. web traffic

The router has a central role in our transparent cache topology. With "policy routing" activated, it diverts TCP port 80 traffic towards a cache server. Although the latency introduced by router is measurable and definite the following considerations prove that it represents only a small fraction of the packets' travel time:

- Cisco routers running IOS 11.3 or later versions perform policy routing in the fast switched path (earlier versions perform it on the slower process switched path). There is no performance degradation when policy routing is configured.
- Our router is a high-end series cisco7505 with fast switching bus and network interfaces.

The latency added by the cache-router network segment is also irrelevant. The average RTT between cache server (10Mbps Ethernet) and clients (100Mbps Fast Ethernet) is 1 ms. Under optimum conditions the

RTT time is very low (0.0004 sec) touching the maximum values of a few ms when the network is loaded. It's a pity that the router's Ethernet port is not full duplex; if it were, the transfer times would be even faster.

The latency introduced by the cache server depends on various factors such as CPU speed, memory size, disk I/O performance, operating system etc. Instead of performing complex data monitoring and calculation we measured only one parameter: the time elapsed between the original client request and that performed by cache server in order to fetch the object directly.
On the assumption that the latency introduced by router and router-cache network segment can be neglected (as described above), that "switching path" in the cache server is nearly as fast that of the router and that Network behavior is unchanged, the measured value represents the latency of a cache server as a whole[4]. The download time of web objects is thus increased by this value with respect to that obtained if clients were to retrieve them directly.
We used two methods of measurement this time: the good old tcpdump and netflow. The data gathered by tcpdump have a major time stamp accuracy (within 4 microseconds) but are less readable[5]. Those collected by netflow are ready for use but flow creation time stamp is expressed in hundredths of a second. The measured values vary between 2 and 70 ms.
If we now compare the average download time of objects retrieved from origin servers (680ms) we deduce that latency introduced by cache server has no impact on service quality and is not noticeable by humans.

3.6.3 Problems encountered

No particular problems were encountered during the test period. One peculiar problem was noted however; while examining the cache log file we found a very large number of requests which generated error code NONE/400 (client bad request). These requests were originated by the Real Player background process that was continuously sending http requests for a non-existent URL (to refresh headlines). The trouble was corrected by installing the new version of the product.
The transparent cache server breaks IP address-based authentication by hiding the real identity of the client. In that case the traffic coming from authorized clients must be routed directly to the origin server. The data flow of these client-server pairs must be excluded from the traffic redirection performed by router.

3.6.4. Impact of this approach on the final user

In spite of some initial resistance, the service was widely accepted. A few users had worried that their network transactions could be controlled, but this was easily overcome by guaranteeing that log files would be anonymized and IP address of stations hidden.

3.7. Future work

This paper outlines results from the first step of a work in progress. For the next step we plan:

- to extend the experimentation to a wider user community, i.e. the CNR Research Area in Pisa, covering a wider variety of thematic interests.
- To add the cache to the CNR hierarchy.
- To deploy the WCCP protocol in order to obtain automatic traffic redirection and fail-safe service.

4. CONCLUSION

This paper describes the practical experience of introducing transparent web caching in our environment. Our primary goal was to experiment and evaluate this technology in a production network. Although the IAT user community is small and homogeneous, which affected some of the measured variables, the data analysis and users' feedback has confirmed the service's utility. The same outcome, in terms of bandwidth savings and latency, was obtained at the CNR Research Area, located in Bari. Unlike Pisa, their user community is much larger resulting in 600MB of http traffic per day.

The results show that:

1. Quality of service improved with 42% of Internet objects served from the cache. The latency and downloading time of web pages was reduced resulting in perceivable service improvement. No noticeable quality of service variation was observed for objects downloaded from origin servers.
2. A small but important bandwidth savings was achieved, corresponding to 20% of the byte hit rate.
3. It reduced unnecessary traffic sent to the Internet, thus diminishing the load on popular web servers.

4 This is true for persistent http connections only. With http/1.0 each separate http connection is delayed by this time value.
5 We used EtherPeek of ag group,inc to facilitate analysis of data obtained by tcpdump.

REFERENCES

[DILL00] J. Dilley, I. Cooper. Known HTTP Proxy/Caching Problems - http://www.ietf.org/internet-drafts/draft-ietf-wrec-known-prob-01.txt, September 8, 2000.

[PAC96] Navigator Proxy Auto-Config File Format, http://home.netscape.com/eng/mozilla/2.0/relnotes/demo/proxy-live.html, March 1996.

[GAUT99] Paul Gauthier, Josh Cohen, Martin Dunsmuir, Charles Perkins. WPAD - http://www.wrec.org/Drafts/draft-ietf-wrec-wpad-01.txt, June 1999

[CIES99] M Cieslak, D Forster. Web Cache Coordination Protocol V1.0, Internet draft, June 1999 http://www.wrec.org/Drafts/draft-ietf-wrec-web-pro-00.txt

[CEUG] Cisco Cache Engine User Guide, Version 2.1.0 http://www.cisco.com/univercd/cc/td/doc/product/iaabu/webcache/ce21/ver21/

[CIES99] M Cieslak, D Forster, G Tiwana, R. Wilson. Web Cache Coordination Protocol V2.0, Internet draft, 13 Jul 2000, http://www.wrec.org/Drafts/draft-wilson-wrec-wccp-v2-00.txt.

[WESS]Duane Wessels. SQUID Frequently Asked Questions - http://squid.nlanr.net/Squid/FAQ/FAQ.html

[BEER00] Cord Beermann. Calamaris - http://Cord.de/tools/squid/calamaris/, Oct 2000.

[OETI00] Tobias Oetiker Dave Rand. Multi Router Traffic Grapher. http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/mrtg.html

[GRAB99] Matija Grabnar. SQUID and MRTG: to SNMP or not SNMP? Feb 1999 http://www.terena.nl/d2-workshop/d2cache2000/lecture.html

[CFLO] CAIDA Cflowd, http://www.caida.org/tools/measurements/cflowd

[DUSK97] B.M. Duska, D. Marwood, M. J. Feeley. The Measured Access Characteristics of World-Wide-Web Client Proxy Caches - USENIX Symposium on Internet Technologies and Systems, Dec 1997

[FIEL99] R. Fielding, UC Irvine, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. RFC 2616 - http://www.ietf.org/rfc/rfc2616.txt, June 1999

[WILL99] Craig E. Wills and Mikhail Mikhailov. Examining the Cacheability of User-Requested Web Resources. WCW99, San Diego, USA.

APPENDIX A

HTTP protocol

The original HTTP protocol was not designed to include caching in the client-server request/response chains and has been extended to include basic cache mechanisms. The HTTP/1.1 protocol offers [FIEL99]:

- Implicit directives such as server-specified expiration times and validators; these mechanisms are used for object validation, in order to reduce unnecessary traffic.
- Cache-Control header containing explicit directives to the HTTP caches (set by http server or client). These directives typically override the default caching algorithms. Cache-control directives include: restrictions on what is cacheable (public, private, no-cache) imposed by the origin server; restrictions on what may be stored by a cache (no-store); modifications of the basic expiration mechanism (max-age, s-maxage, max-stale, etc.); controls over cache revalidation and reload (only-if-cached, must-revalidate, etc.), imposed by a user agent.
- General header fields including information for cache systems. Inclusion of If-Match, If-Modified-Since fields, If-None-Match or If-Unmodified-Since header field makes the GET method conditional; that is, the entity is transferred only if the requested condition is matched. If this condition is not satisfied, the server returns only the entity metadata. In this way unnecessary traffic is avoided and cache information could be efficiently updated. The Pragma header is included for compatibility with the protocol v. 1.0. The pragma: no-cache header requires one authoritative copy of the resource. This feature directive permits the client to refresh cached copy that has become corrupted or stale.

The HTTP/1.1 specification allows a proxy cache to switch over to tunnel mode when it receives a request carrying method it does not understand how to handle. This makes protocol extensible.

In order to benefit from the new HTTP extension, a full implementation of HTTP 1.1. features and mechanisms in client, server and proxy cache is required. Inaccuracy in the protocol implementation could generate data inconsistency.

In addition, the lack of cache directives in http response cause a partial loss of the potential reuse of Internet objects, since cacheable objects are not cached [WILL99].