Web Distribution Systems : Caching and Replication

Nikhil Chandhok, *chandhok@cse.wustl.edu*

Abstract

This document gives an overview of the current caching and replication terminology. It describes the current protocols and is a starting point for anyone interested in the subject. It starts by describing various caching and replication systems and then moves onto a brief description of most of the relevant protocols. It finishes by giving a brief product description of some of the products available in the market.

Other Reports on Recent Advances in Networking
Back to Raj Jain's Home Page

Table of Contents:

1.0 Introduction

The following subsections introduce the concept of caching and replication .

1.1 Overview of Web Caching

The Internet phenomenon happened just a few years ago . In a short span of time the Internet has become the source of a whole lot of information and the number of people getting added to the World Wide Web has increased exponentially. It has become a central focus for electronic commerce, information services and education . This has lead to increased business opportunities for everyone involved in the business of providing Internet services. However this sudden growth has some growing pains of its own. Everyone with a stake in the business of building the Internet of tomorrow - content providers, network service providers , telecommunication companies , hardware manufacturers , software developers are working under the strain of constant growth . This strain leads to increased latency over the Web . Latency can be for many reasons . Servers can get overloaded due to increased traffic especially when there is a sudden surge in request for a particular type of resource . Networks can get congested due to increased traffic and this very common across transoceanic links that often cost millions of dollars per month .

In an effort to alleviate this growth induced pain the networking industry has come up with various ways . A common way to do this is to upgrade the loaded resource : a faster server , a bigger switch , reengineering the network . However this approach is not always economically feasible and more importantly , it also fails to consider the numerous parties involved in a single Web transaction . In addition to the user and the information provider , several ISP's and exchange points participate in the exchange of information . A transaction is only as good as the weakest network link .

Caching has proven useful tool in reducing end user latency in the Web .

- Caches diminish the need for network bandwidth , typically by 35% or more , reducing the traffic from browsers to content servers .
- Caches can improve Quality of Service (QOS) for browsers users by delivering content at higher bandwidth and reducing latency .
- Caches gather better and superior network management information and allow for smarter network management .

Back to Table of Contents

1.2 What is a Web Cache ?

A cache is a temporary storage location for copied information . There are over a billion pages(or objects) on the internet. Many users request the same popular objects . An example of that would be the top logo image of Yahoo.com which appears in almost all Yahoo pages . The image must be delivered to the browser each time the browser accesses any of Yahoo's pages an these pages are requested a number of times each day by different users .

A Web cache is a dedicated computer system which will monitor the object requests and stores objects as it retrieves them from the server . On subsequent requests the cache will deliver objects from its storage rather than passing the request to the origin server . Every Web object changes over time and therefore has a useful life or "freshness" . If the freshness of an object expires it is the responsibility of the Web cache to get the new version of the object . The more the number of requests for the same object the more effective will the Web cache be in reducing upstream traffic and will also help reducing server load, resulting in less latency.

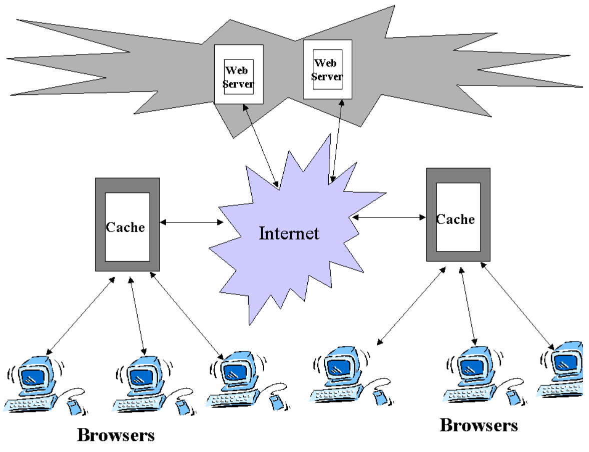

fig 1: Basic Architecture

The most basic architecture of a web caching system is shown in fig 1 . Caching systems are deployed at network boundaries typically at entry points to a network. These caching systems will monitor traffic that goes out through them and perform the caching functions as explained above .

Back to Table of Contents

1.3 Replication

Replication is a technique similar to caching, but is generally considered to be more active. The process of replication copies cache content and pushes it on to one or more cache servers across the network. Replication is required to distribute objects among the servers to maintain the freshness of content across servers, which results in reduced upstream network traffic. Typically the same content is pushed across several machines making it more efficient to use Multicast. Replication is critical in global operations, where cost of international traffic is high and ways have to be found to mirror data without using too much bandwidth.

Back to Table of Contents

1.4 Caching And Replication : Some Common Terms

This section discusses the meaning of some of the terms that are going to be used in the rest of this paper.
Client
A client is any application which establishes connections for the purpose of sending requests.
User Agent
The client which initializes a request. These are often browsers, editors etc.
Server
An application which will accept connections for the purpose of responding to the requests by sending back responses. Any given program can either be a client or a server, depending on the type of role a particular connection performs. The term should be used with reference to each connection rather than the whole program. A server can be an origin server or a gateway or a tunnel or a proxy based on the type of request .
Origin Server
The server on which the original content resides or is to be created .
Proxy
An intermediary system which will act as both a server and a client for the purpose of making requests , on behalf of other clients . Normally proxies sit at entry points into the network. Requests may be served internally or they can be passed on. If the requests are passed on then the proxy will act as the client while getting information from the origin server and then pass this information on to the client which requested it initially .
Reverse Proxy
Reverse Proxy is a server side proxy , which resides on server side and will cache objects frequently requested by the clients . Reverse proxies are often used as server side portals through network firewalls and as helper applications for off loading requests from origin servers.
Caching Proxy
It is a proxy with a cache, acting as a server to clients and client to servers.
Cluster
A tightly coupled set of devices acting together to share load.
Local Caching Proxy
The caching proxy that the user agent connects to.
Intermediate caching proxy
From the content providers view all the caches participating in the caching mesh that are not user agent's local caching proxy .
Cache Server
A server which will serve requests made from local and upper level caching proxies but won't serve as a proxy itself .
Cache Array, Diffused Array, Cache Cluster
A cluster of caching proxies acting logically as one service and partitioning URL name space among themselves.
Caching Mesh
A loosely coupled set of co-operating proxy caches or caching servers or clusters acting independently but sharing content between themselves using Inter Cache Communication Protocols .
Transparent Proxy
A transparent proxy is a proxy which removes the necessity for browser configuration when the user accesses a proxy .
Hot Spots
A sudden surge in traffic, at a server, when a particular resource is being tried to access.
Proxy Discovery
This describes the discovery and configuration for use of a proxy in an environment where the content consumer may be unaware of the proxy's existence. The use of the proxy is transparent to the content consumer but not to the client.
Appliance
A dedicated piece of hardware serving as a cache.
Replica Origin Server
Origin server storing a persistent replica of a data set stored at the authoritative reference.

Back to Table of Contents

2.0 Types of Caching and Replication Systems

The following sections will discuss the various caching and replication mechanisms. Caching and replication methodologies are discussed in this section .

2.1 A diagrammatic representation

fig 2 describes the components that make up a Web caching and replication system, with communication between the components.

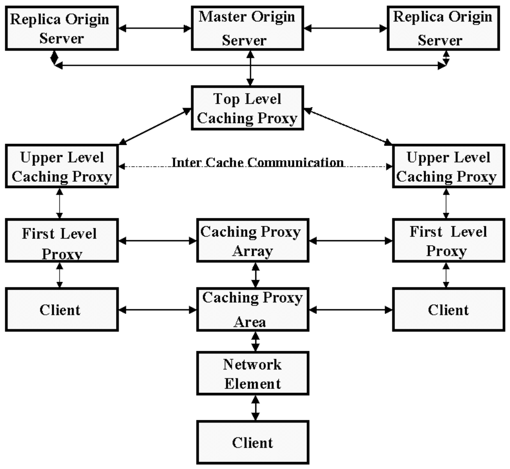

fig 2: The Overall Picture

Back to Table of Contents

2.2 Proxy Caching

Proxy servers are typically deployed in networks at entry points into the network and in many cases to prevent undesirable traffic from entering the network and to monitor the traffic that goes out of the network . Because of its location, it makes sense to deploy caching functionality on proxy servers. Hence the name proxy caches .

Proxy caches help enterprise networks and Internet Service Providers realize the benefits of caching by sharing a cached object among multiple objects . When the network user requests an object the request is first sent to the proxy. If the proxy has the requested object, then the proxy sends the object to the requesting client . If the requested object is not at the proxy, then the server will generate a new request to the server and on behalf of the client , retrieve it from the server , store it locally and send a copy back to the requesting client . When a different client requests the object, the server can satisfy the request locally . The key is that if a user requests a particular object then it is available to the rest of the network users until such time that the "freshness" of the object expires .

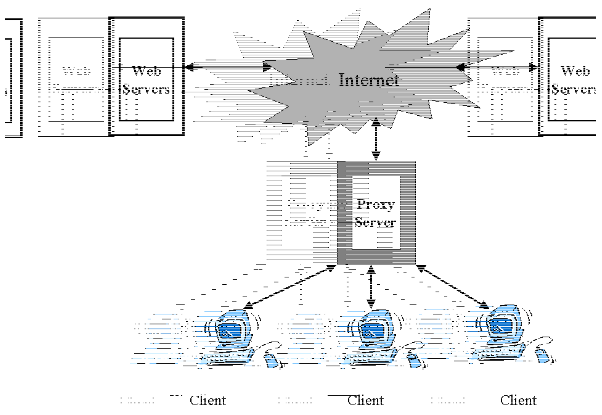

fig 3: Proxy Caching

Back to Table of Contents

2.3 Transparent Proxy Caching

Traditional proxy caching requires manual configuration of the browser so that the browser or any other user agent used knows the address of the proxy . It would be better if we could intercept Web traffic without configuring each browser. This transparency means that the user need not be aware of the proxy.

A Web cache is said to be transparent if clients can access the cache without the need to configure the browser, either using a proxy auto configuration URLor manual proxy configuration. Transparent caches fit seamlessly into the current network infrastructure. They function more like transparent firewalls rather than a set of discrete proxy servers. It makes business sense to incorporate transparent caches into the network because of the low overhead involved. However once incorporated into the network the transparent cache should be as scalable and and fault resistant as the rest of the network.

An HTTP request is to be redirected into a proxy cache in different ways. The most popular way is to use a L4 switch. The L4 switch will trap IP packets with a destination port of 80 and forward the packets to a proxy cache. HTTP request requires that a TCP connection be established and the first packet be a SYN packet. The L4 switch will forward this packet to the proxy cache. The proxy cache will act as the server and return a SYN/ACK packet corresponding to the origin server's IP instead of its own. The client will think that the packets are coming from the origin server and the operation of the proxy cache will continue as described in the paragraph above.

In case of transparent caching some other proxy cache which lies between the current proxy cache and the origin server may intercept requests from the current proxy cache. Therefore when the client finally gets the requested object, the client does not know whether it is the origin server or the proxy cache or some other intermediary proxy cache which satisfied the request because the IP header in the response always contains the origin server's IP address. The proxy cache may or may not lie in the direct path between the origin server and the client . If it doesn't then, in case of cache failure, providing that the switch is monitoring the caches, the switch can redirect traffic direct to the server . It is also possible for the switches to do load balancing , by distributing the traffic among the web servers . This method of transparent caching requires that a protocol run between the switches and the cache or caches .

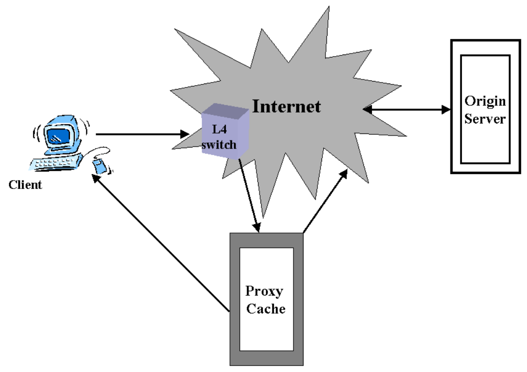

fig 4: Transparent caching

Back to Table of Contents

2.4 Hierarchical Web caching

Proxy caching underscores the point that sharing is useful . But every cache has a finite size and there is a limit to the number of objects that can be cached. A single cache will reduce the amount of traffic generated by the clients behind it. Similarly a group of Web caches can benefit by sharing a particular Web cache. Therefore caches have been implemented hierarchically as shown in fig 5. In a cache mesh or a cache hierarchy one cache will establish peering relationships with other caches in the vicinity. There are two types of relationships : parent and sibling. A parent cache is essentially one level up in a cache hierarchy. A sibling cache is on the same level. The terms "neighbor" and "peer" are used to refer to either parents or siblings which are one hop away.

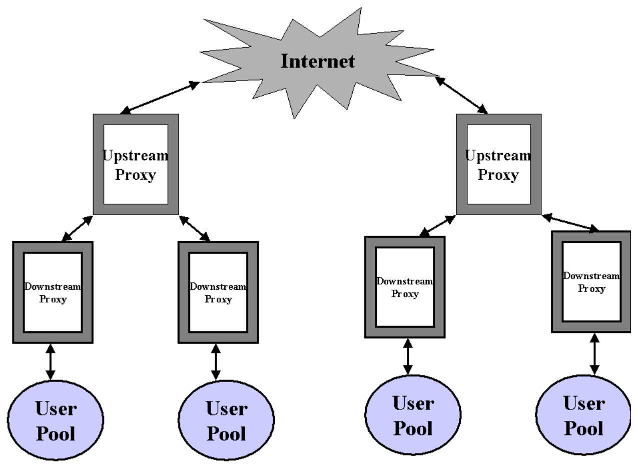

fig 5: Hierarchical caching

The general flow of documents is up the cache hierarchy . When a cache does not hold a particular object, it may ask via ICP[explained later], whether any of its ICP neighbors has the object. If any of its neighbors has the object ["neighbor hit"] then it retrieves it from that particular cache. If none of the neighbors has the object ["neighbor miss"] then the cache must forward the request either to a parent or directly to the origin server . The parent will query recursively until such point that the object is found . The essential difference between the parent and the sibling is that a "neighbor hit" may be fetched from either one but a "neighbor miss" may **only**be fetched from a parent . In other words, in a sibling relationship, a cache can only ask to retrieve objects that the sibling has already cached whereas the same cache can ask a parent to retrieve any object regardless of whether or not it is cached or not .

Squidallows complex cache hierarchies. For example, one could specify that a given neighbor be used for only a certain class of requests, such as URLs from a specified DNS domain. Additionally it is possible to treat a neighbor as sibling for some caches and a parent for others.
The cache hierarchical model described here includes a number of features to prevent top level caches from becoming choke points. One way is to restrict parents by domains. Another way is to make sure that caches forward cacheable requests to its neighbors. A large class of Web request are unreachable examples of which would be: database queries, authentication, session encrypted data. Lower level caches should recognize such requests and make sure these are not passed onto the parent caches.

Back to Table of Contents

2.5 Distributed Caching

Traditional cache hierarchies implemented have shown the following problems as mentioned in [Tewari98]:

- A request may have to travel many hops in a cache hierarchy directory to get to the data , and the data may traverse several hops back to get to the clients .
- Second , cache misses will be significantly delayed by having to traverse the hierarchy .
- Little sharing of data among caches .
- Shared higher level caches may be too far away from the client and the time for the object to reach the client is just unacceptable.

To overcome these drawbacks we introduce distributed caching which allows the distribution of caching proxies geographically over large distances and attempts to overcome some of the drawbacks of traditional hierarchical caches. Caches are organized into cache clusters with no definite hierarchy among them as shown in fig 6. This it achieves by using optimizations and allowing the definition of complex inter cache relationships .

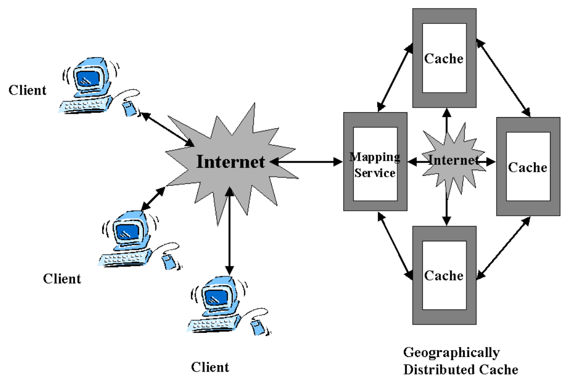

fig 6: Distributed Caching

Distributed Caching several benefits including :

- Distribution of server loads , with downstream proxies , offloading cache hits from upstream proxies .
- Improvement of client performance by bringing caches closer to the client .
- Improve in cache hit rates due to increase in cache space .

[Tewari98]describes the following strategies that can be used while designing distributed web caches :
The first strategy has 3 salient features :

- Separation of data and metadata paths and maintaining a hierarchy of metadata paths that tracks where copies of data have been stored .
- Maintaining location hints so that caches can locate nearby copies without much data
- Using direct cache to cache data transfers to avoid store and forward delays

The use of location hints allows the cache to locate a neighboring cache with a copy of the data and initiating a cache to cache transfer without going through the complete cache hierarchy. Caches in the system will detect misses locally using hints and proceed directly to the server. A meta-distribution hierarchy is used to distribute hints effectively and stores data at the leaves of the hierarchy .

The second strategy uses *push caching* wherein data is pushed near the clients by making an informed guess as to what the clients may access in the near future. Thus it removes the compulsory miss penalty that is always experienced when the client first accesses the object . Please refer to [Tewari98]for further details.

Back to Table of Contents

2.6 Replication using Multicast

Replication uses Multicast to push data across to mirror origin servers. Replicating data and distributing it as needed is called mirroring. Pure replication differs from caching in the sense that caching systems "pull data" from the origin server, while replication systems tend to "push" data to maintain mirror copies of the same data at various place on the network. Multicast can either be best effort or reliable. Best effort uses UDP/IP Multicast, while reliable Multicast needs additional protocol machinery in place to recover from lost packets. Best effort Multicast simply takes advantage of IP Multicast in disseminating objects. However objects are stored in remote machines only if the object is still intact. Checksums can be used to verify if the objects delivered are intact. If the objects are intact they can be saved while as non intact objects are discarded. Best effort Multicast though a lightweight protocol can fail when the error rate is high, where if a particular packet is dropped the whole object may have to be dropped. Reliable Multicast like in [Sanjoy99]on the other hand uses UDP/IP Multicast for best effort dissemination, but uses ACK/NACK based protocol to detect losses lost packets and retransmit them. LSAMexplained later uses the concept of definite Multicast channels which allow Multicast only to a set of serving clients instead of the all the clients. CRISPis another replication system explained later in the report.

Back to Table of Contents

2.7 So what's the difference between the Caching and Replication ?

Caching systems :

- reduce network latency by bringing content closer to the content consumer.
- are essentially reactive wherein a data object is cached only when the client requests it.
- meet traffic reduction goals by only getting content when requested.
- have consistency problems due to their reactive nature
- can have reliability problems as they are normally placed at network entry points and a cache failure may sometimes bring the whole network down.

Replication systems :

- know exactly when a object changes and push the objects immediately .
- ensure content freshness due to their reactive nature.
- have very high fault tolerance due to replication of data, which ensures that even if a web server goes down requests can be redirected to another origin server.
- knowledge of the persistent domain allows load balancing .
- consume more disk space .
- need efficient algorithms for load balancing .
- may increase network traffic if Multicast is not used judiciously .

Caching and replication are used in a mixed environment and the discussion of one is not complete without discussing the other. The following sections make this more clear.

Back to Table of Contents

3.0 Replication Relationships

This section defines the types of replication relationships. It also discusses some of the protocols used to implement the same .

3.1 Client to Replica relationships

This section will describe the cooperation and communication between clients (both user agents and proxy caches) and the replica origin servers . These are used to discover an optimal web origin server. A number could define optimality. Normally it is based on proximity.

3.1.1 URL Redirection

Description :
A simple and commonly used mechanism to connect web clients with origin servers replicas is to use the HTTP protocol response code 307 Temporary Redirect. The initially connected replica may decide to connect to the services or decide to redirect the client to the proper replication server. Further details are available in section 10.3.8 in HTTP 1.1 RFC 2616.
Security:
Relies on HTTP security entirely .

3.1.2 DNS redirection

Description:
DNS provides a more sophisticated client to replication policy . DNS servers implement order of addresses based upon quality of service policies . When a Web client resolves the name of a web server DNS can be used to order the IP addresses of web servers starting with the most optimal replica and ending with least optimal replica . The DNS however has limited control on the request reaching the appropriate replica . Many intermediate servers tend to cache the URL to IP translation . Moreover some client browsers typically cache some address translation . The choice of the most optimal replica changes with time . Therefore the DNS system specifies a validity period for caching the result of the logical name resolution. DNS also provides load balancing capabilities by distributing requests to the most optimal replica . A detailed description can be found in RFC1794.
Security :
Relies entirely on the DNS security .

Back to Table of Contents

3.2 Replica to Replica Relationships

This section will deal with inter replica communication and the method replica severs follow to replicate data sets among themselves and the origin servers . For further details refer [DraftTax].

3.2.1 Batch Drive Mirror Replication

Description:
In this model the replica server that needs to be updated will initiate communication with the master origin server. The communication is established at intervals based upon queued transactions which are deferred for processing . The scheduling policy can be based upon a number of factors ,but they have to reoccur over a period of time. Once communication is established data sets are copied to the initiating replica web server.
Security :
Relies on the security on the protocols that are used to initiate communication and transfer data. FTP and RDIST are the most widely deployed protocols .

3.2.2 Demand Driven Mirror Replication

Description:
In this model the replica acquires the content as needed on demand. This is normally followed in reverse proxies , where in a client may request a document and when it can't find it with itself , it initiates communication with the master origin server to get it.
Security :
Relies on the protocol that is used to transfer the URLs . FTP, GOPHER ,HTCP, and ICP are the most commonly used protocols .

3.2.3 Synchronized Replication

Description:
In this model of replication origin servers will cooperate using synchronizing strategies based on specialized replication protocols to keep replica sets coherent . Synchronization strategies range from the tightly coherent where the synchronization takes place every few minutes to loosely coherent where the synchronization takes place every few hours . Normally the synchronization follows a coherency model designed to reduce network bandwidth required to replicate.
Security:
All of the known protocols follow strong cryptographic key exchange methods either based on the Kerberos model or the public/private key RSA model.

Back to Table of Contents

4.0 Caching Relationships

The following sections discuss caching relationships and some of the current protocols used to implement these relationships.

4.1 Client Cache Communication

As the name implies Client Cache communication deals with ways in which clients can find the caches. The following section will discuss various existing protocols the clients can use to discover a cache. Starting of from the most basic we discuss some of the better protocols.

4.1.1 Manual Proxy Configuration

Description
Though this can't exactly be termed as a protocol, yet it is important to include this in the current discussion. As the name implies each user will manually configure its web clients and it requires that the user have the knowledge of the proxied protocols and local policies.
Security
The potential for going wrong is high as each user will individually set preferences.

4.1.2. Proxy Auto configuration [PAC]

Description:
A Javascript page on a web server hands out information on where to find proxies . Clients need to point at the URL of this page. No manual configuration will therefore be necessary. This script will reside at a certain central URL from where it can be accessed by all clients. However users still have to configure this URL into their browser or Web client .
Security :
One definite advantage of PAC is that administrators can update the proxy configuration without user intervention. Therefore it allows implementation of a common policy across the organization .

Back to Table of Contents

4.1.3 Web Proxy Auto Discovery [WPAD]

Description :
For a detailed description refer to [WPAD]. Web clients implementers are faced with a dizzying array of resource discovery protocols at varying levels of implementation and deployment . This complexity hampers deployment of a "web proxy auto discover facility". WPAD does not define any new standard. Instead WPAD uses existing pre-existing Internet resource discovery mechanisms to perform discovery of proxy. Thus the WPAD protocol reduces to providing the web client with a mechanism to discover the PAC URL. WPAD does not specify which proxies will be actually used. The PAC script will decide that.
The WPAD protocol specifies the following

- how to use each mechanism for the specific purpose of web proxy auto discovery

- the order in which the mechanisms should be performed
- the minimal set of mechanisms which must be attempted by a WPAD

The resource discovery mechanisms utilized by WPAD are as follows.

- Dynamic Host Configuration Protocol .
- Service Location Protocol
- "Well Known Aliases" using DNS A records.
- DNS SRV records .
- "service: URLs" in DNS TXT records .

Security:
Relies upon DNS and HTTP security .

Back to Table of Contents

4.1.4 HTTP

Description:
HTTP is used for communicating with a web proxy . A web proxy may also support other protocols. HTTP 1.1 added support for cache directives .These cache directives provide information on cacheability of web objects, time to live, as well as other information for the cache to use. The privacy issue goes both ways with proxies. Proxies may be used to anonymize users users to web servers and they may be used to track every web movement the user makes .
Security:
Redirecting users through a proxy server gives access to all information about usage and content of documents. This may be a security risk, or it may be useful to have a check point for communication as in firewalls. Access controls based on IP numbers are not indicated in HTTP headers , and caches have no way of knowing if a web object was access restricted until such time that additional headers are provided by the origin server .For further details refer [HTTP1.1]

Back to Table of Contents

4.2 Inter Cache Communication

As the name suggests Inter Cache communication will deal with cooperation and communication between caching proxies .We discuss the prevalent protocols and evaluate them .

4.2.1 Internet Cache Protocol [ICP]

Description:
Probably the most widely deployed protocol in Caches world wide and was first used in SQUID. ICP is used by caches to query other web caches about web objects, to see if a web object is present at other caches. To do this it uses UDP. Since UDP is unreliable by design, an estimate of network availability may be calculated by ICP loss. This rudimentary loss measurement does together with round trip times is useful in load balancing caches. ICP provides hints about the location of Web objects in neighboring caches Although it is possible to maintain cache hierarchies without using ICP, the lack of ICP or something or similar prohibits the existence of sibling meta-communication relationships i.e. mechanisms to query nearby caches about a given document.
Security:
ICP does not convey information about HTTP headers associated with a web object. HTTP headers may include access control and cache directives. Since caches ask for objects, and then download the objects using HTTP, false cache hits may occur. Because ICP trusts the validity of an address of an IP packet it is susceptible to IP address spoofing. Security is an issue with ICP over UDP because of its connectionless nature .Some other security concerns would be:

  - Inserting Bogus ICP Queries
  - Inserting bogus ICP replies
  - Cache Poisoning
  - Eavesdropping
  - Blocking and delaying messages

For more details about ICP refer to [RFC2187].

Back to Table of Contents

4.2.2 HyperText Caching Protocol [HTCP]

Description:
Hyper Text Caching Protocol is an experimental protocol which is used to discover HTTP caches and cached data, managing sets of HTTP caches and monitoring cache activity. It can be used to monitor remote caches additions and deletions and sending hints about web objects such as the third party locations of cacheable objects or the measured uncacheability or unavailability of an object. HTCP includes HTTP headers while ICP doesn't. The headers have vital information which can be used by the proxy caches.
Security:
If the authentication in the protocol is not used it is possible to both possible for an unauthorized third party to modify a cache using the HTCP protocol . The HTCP protocol does use the MD5 secret authentication. For further detail refer to [DraftHTCP].

Back to Table of Contents

4.2.3 Cache Array Routing Protocol [CARP]

Description:
CARP is a hashing function based protocol. It divides URL space among a cluster of proxy caches. CARP allows for distributed caching, allowing hierarchical proxy arrays. Basic mechanism of CARP is:

  - All proxy servers are tracked through an "array membership list", which is automatically updated through a time-to-live countdown function .
  - A hash function is computed for the name of each proxy server
  - A hash value of requested URL is calculated .
  - The hash value of the URL is combined with the hash value for each proxy . Whichever URL+Proxy Server hash comes up with the highest value , becomes owner of the information cache .

The result is that an exact location where a requested URL either is already stored locally or will be stored after caching . The hash function allows the distribution of load among the serving caches. CARP doesn't need the maintenance of massive location tables. It requires the browser to run the same math function it is running. Strictly speaking CARP is not an inter cache communication protocol. One of the main goals of CARP is to reduce inter cache communication. CARP is more scalable than ICP since adding a new cache need not necessarily increase traffic as in the case of ICP. Please refer to [DraftInterCache], [CARP97]for more details.
Security:
Not known.
Back to Table of Contents

4.2.4 Cache Digests

Description:
Cache digests are a response to the problems of latency and congestion associated with traditional caching methodologies like ICP and HTCP . Unlike most other protocols Cache Digests support peering mechanisms between cache servers without a request response system . This has been the new research direction in the past year or two . A summary of contents of the server (the Digest) is used to score a hit with a relatively high degree of accuracy . Each proxy keeps a a compact summary of the cache directory of every other proxy . When a cache miss occurs , a proxy first probes all

the summaries to see if the request might be a cache hit in other proxies and sends a query only to those summaries that show promising results [SUMMARY98]. To reduce memory requirements "Bloom filters" are used . A "Bloom filter" is a computationally efficient hash based probabilistic scheme that can represent a scheme a set of keys [In this case URLs of cached documents] with minimum memory requirements, while answering membership queries with zero probability for false negatives and low probability for false positives.
Security:
Cache A can build a fake peer Digest for Cache B and serve it to B's peers if requested. This way A can direct traffic to/from B. The impact of this problem is minimized by the 'pull' model of transferring Cache Digests from one server to another, but this "Trojan horse" attack is possible in a cache mesh. If the contents of a Digest are sensitive secure methods that are used to secure an HTTP connection should be used to transfer data among caches.
Please refer to [DraftInterCache]for more details.

Back to Table of Contents

4.2.5 Multicast based protocols

Multicast based caching and replication is still in experimental stage. We discuss two such projects.

4.2.5.1 Large Scale Active Middleware [LSAM] Proxy Cache

Description:
The LSAMproxy cache (LPC) provides a distributed cache system that uses Multicast for automated push of popular web pages. LPC uses proxy caches that are deployed as a server pump and a distributed filter hierarchy as shown in fig 7 . These components automatically track the popularity of web page groups, and also automatically manage server push and client tuning. The system caches pages at natural network aggregation points, providing the benefits of a single, central shared proxy cache, even where no such central proxy could exist. The server pump monitors the server for affinity groups. It will create and destroy Multicast channels as different affinity groups more or less popular. The filter cache is closer to the client and has a partitioned cache to allow multiple channels. Individual request trigger Multicast responses when pages are members of active affinity groups. A request is checked at intermediate proxies and forwarded through to the server, as in conventional proxy hierarchies. The response is Multicast to the filters in the group by the pump and unicast from the final proxy back to the originating client. Subsequent requests are handled locally from the filters near the clients.

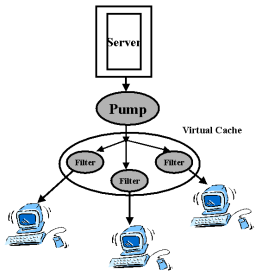

fig 7: Pump and filter relationship , as in [Touch99]

The system has an initial prototype implementation, and it is evolving to mitigate the effects of a deep proxy hierarchy, needed for the Multicast self organization. LPC reduces overall server and network load, and increases access performance, even for the first access of a page by a client.
Security:
LSAM uses a strong security transaction mechanism. It uses digital signature and encryption services as applicable to MIME objects.

Back to Table of Contents

4.2.5.2 CRISP

Description:
The CRISP project seeks to build more effective distributed Web caches by exploring alternatives to the hierarchical structure and Multicast handling of probes common to the most popular distributed Web cache systems. CRISP caches are structured as a collective of autonomous Web proxy servers sharing their cache directories through a common mapping service that can be queried with at most one message exchange. A caching server probes the cache with at most a single unicast message exchange to a mapping server, which may be collocated with the caching server. If the probe is a HIT, the object is fetched directly from the peer that caches it. If the probe is a MISS, the requesting proxy fetches the object directly from the Web site. Individual servers may be configured to replicate all or part of the global map in order to balance access cost, overhead and hit ratio, depending on the size and geographic dispersion of the collective cache.
Types of directory structures that CRISP follows are:
Partitioned Synchronous Directory
In this structure, the mapping service consists of several mapping servers, each assigned some exclusive subset of the URL space using a hash function in a manner similar to the Cache Array Routing Protocol (CARP) .Caching servers synchronously query the assigned mapping server when a client's request results in a local cache miss, and notify the appropriate mapping server when an object is added to or evicted from the local cache.
Replicated Asynchronous Directory
In geographically dispersed CRISP caches, the network latency to query any shared mapping server will be unacceptable to some proxies, regardless of where that mapping server resides in the global cache.
CRISP implements variants that asynchronously replicate the global directory. Replicas can be placed closer to a group of caching servers, or even collocated with each caching server. In a sense, the replicated directory structure is the opposite of the hierarchical ICP caches: rather than Multicasting queries and generating no updates to the map, this structure Multicasts the updates and generates local queries. The key advantage of Multicasting updates over Multicasting queries is that the updates to the map can be propagated asynchronously, whereas queries are by their nature synchronous.
Replicated Partial Directory
Replicated Partial Directory seeks to further reduce the cost of replicating the cache directory, but without compromising hit ratios significantly. The idea is to identify a subset of the cache contents that is most likely to yield productive hits, and replicate only those entries (called the submap) at each mapping server. Note that partial directory replicas can be used to supplement rather than replace central directories. One approach is to check the submap replica first to speed up the majority of hit cases, but fall back to querying a central mapping server for objects that miss in the submap. This approach delivers hit ratios and miss latencies comparable to the synchronous directory configurations, but with lower average hit ratios.
For further details refer [CRISP].
Security:
Not known .

Back to Table of Contents

4.3 Reverse Proxy Caching

Cache engines are frequently deployed nearby clients to ensure faster network response time and minimal WAN bandwidth usage. Thus, caches are caching the clients' most frequently accessed content. In addition, caches can be also deployed in front of Web server farms to increase server capacity. This configuration is called reverse caching since cache engines are only caching content from the servers that they are front ending . This feature is particularly important when cache engines are front ending server farms in which certain content is dramatically popular than other content . Using reverse proxy allows administrators to prevent a small number of high demand URLs from impacting overall server performance . Another benefit of this is that it is not necessary to identify high demand URLs , manually indicated or managed independently from the bulk of URLs on the server. fig 9 shows a many to many reverse proxy caching proxy relationship . There is no dedicated one to one relationship between the server and the client. Rather there is a many to many relationship. Reverse proxy allows load balancing in addition to ability to scale dynamically.

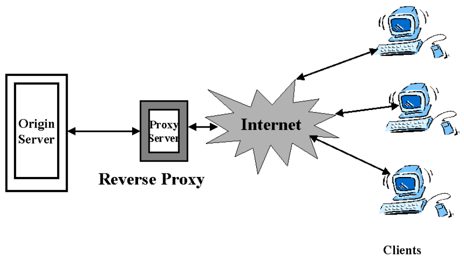

fig 8: reverse proxy caching

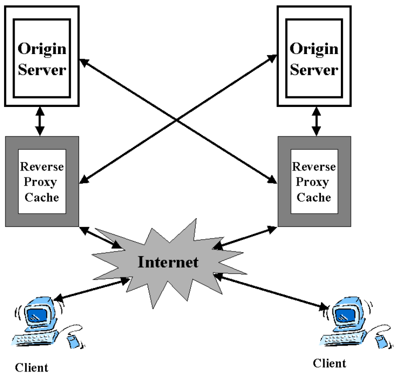

fig 9: many to many reverse proxy caching

Distributed reverse proxyis a variation of reverse proxy caching . Not only does it have a many to many server/cache relationship , but also has a geographic distribution of caches . This can help overcome delays due to congestion or routing over a long distance .

Back to Table of Contents

5.0 Network Element Communication

This section will describe the cooperation and communication between caching proxy and network elements. Examples include routers and switches and are generally used for transparent caching and or diffused arrays. Network elements have the following desirable properties:

- They can be managed like network equipment .
- Designed like networking hardware , resulting in better physical integration into the network , minimizing cost associated with leasing rack space .
- Adapt to conditions or changes in the network , resulting in minimized deployment/operational costs and greater content availability .
- Cut-through routing when a cache fails
- A flexible hashing algorithm
- Ability to limit the number of HTTP flows that it will divert into any particular cache
- Support for access control lists so that caching can be disabled for particular Web servers and
- Ability to detect cache failure by polling the cache for a URL via HTTP
- Ability to detect and avoid routing loops possible with Web cache redirection that occur between
- The switch, the routing equipment, and the caches
- Administrative security
- Redundant switch failover deployment
- Ability to load balance to caches on different subnets

5.1 Web Cache Coordination Protocol

WCCP is a proprietary protocol from CISCO which runs on their Cache Engine 500 series products .
The WCCP protocol is used to :

- associate a single router with one or more web caches for the purpose of transparent redirection of HTTP traffic .
- allow one of the web caches to dictate how the router distributes transparently - redirecting traffic across the associated web caches .

A web cache can dynamically add itself to the server farm by advertising itself to the server farm . The source IP of the originating web cache is used to uniquely identify the server. WCCP is very scalable . It uses a 256-bucket redirection hash table which is stored on a router. Any new cache engine that is added is allocated buckets from this hash table . Whenever a new IP packet comes the IP address is used to index into the 256 bucket hash table . The indexed bucket indicates to which web cache the packets should be redirected. Hot cache engines can be inserted into a fully operating cluster. Periodically the router advertises a list of proxies it considers usable. This information is provided for the designated web caches . WCCP allows dynamic "healing" in case a particular cache cluster fails . The router also makes sure that HTTP traffic directed from members of the cache farm to the origin server is not redirected .
Security :
WCCP v1 does not provide any security feature.
Refer to [Cisco99]for further details.

Back to Table of Contents

5.2 Transparent Proxy Agent Control Protocol

Description:
TPACT runs between a networking element and , functioning as a redirecting element and out of path transparent proxy caching policies . The protocol is similar to WCCP . It allows one or more caching proxies to register themselves with a single network element, to receive redirected web traffic. All of the participating caching proxies unlike WCCP operate as a quorum in dictating web

distribution of traffic.
Security:
MD5 is optionally employed for authentication. Sequence numbers are employed as security against replay attacks .

5.3 Sample products

This section will essentially discuss some other commercial switches that support transparent caching . Refer to [NetCache99]for further details
5.3.1. Alteon L4 Switch

The Alteon AceSwitch family of Gigabit Ethernet server switches with optional L4 switching features can switch up to a gigabit of TCP traffic. Upon cache failure, the AceSwitch can either load balance to another cache or cut the traffic for that cache directly to the WAN. The AceSwitch selects one of the available caches using a hashing function based on the destination IP address of the requested web server. Alternately, it can select a cache using one of several load-balancing algorithms, including least loaded and round robin .With an AceSwitch, the cache need not implement any active failover mechanism.

5.3.2. Foundry ServerIron L4 Switch

The Foundry Networks ServerIron switch provides Server Load Balancing features and can switch up to a full gigabit of HTTP traffic to the cache cluster. The ServerIron switch provides multiple mechanisms for hashing clients-to-NetCache servers. Using bit masks, the hashing mechanism can be keyed on the source and destination IP addresses and port numbers or only the destination IP address (so as to best exploit persistent connections). The ServerIron switch uses physical link, ICMP (ping), TCP SYN-ACK, and HTTP mechanisms for detecting cache failure and re-mapping to a different cache or cutting through directly to the Internet. Foundry's ServerIron switches can be deployed in failover pairs and can also provide full-duplex connection to groups of cache servers through downstream switches.

5.3.3. ArrowPoint "L7" Switch

The ArrowPoint Content Smart Switch products, like the Alteon switches, can switch over a gigabit of TCP traffic, partition traffic based on a hash of the requested Web server's IP address, and recover from a failed cache by cutting through that cache's traffic. The ArrowPoint switch supports a more sophisticated hashing capability than the Alteon. While it can perform the L4 switching , it can also partition client traffic based on the requested URL. To do this, it must accept the TCP connection from the client and handle the HTTP request to obtain the URL " an application layer (layer 7 or L7) function. It then hashes the URL, opens a connection to the selected cache, and finally dispatches the request. The ArrowPoint can even bypass the cache entirely for obviously non-cacheable URLs, such as CGIs and URLs carrying cookies.

Back to Table of Contents

6.0 Product Comparison

Caching solutions is a growing market and the number of product options available to anyone going in for a caching solution is mind boggling. A survey of the available products in the market would merit a paper of its own. However the following table gives a list of some of the caching solutions available in the market. For further details refer to [Reardon99].

Vendor/ **Product /** **Type**	**Configuration**	**Simultaneous Connections**	**Objects Served per second/ Caching Throughput**	**Caching/ Cache deployment (transparent, proxy)/ Security**	**Cached Traffic/ Cache Protocols/ number per cluster**
CacheFlow Inc , www.cacheflow.com /CacheFlow 100 CacheFlow 500 CacheFlow 1000 CacheFlow 2000/ Appliance	Cacheflow 100: 1 10/100-Mbit/s Ethernet port; Cacheflow 500:3 100: 1 Ethernet ports; Cacheflow 1000:2 to 4 10/100 Mbps, Ethernet ports; Cacheflow 2000: 4 10/100 Mbps, Ethernet ports;	CacheFlow 100:1500 CacheFlow 500:5900 CacheFlow 1000:5900 CacheFlow 2000:18500	1000/ CacheFlow 100:1 Mbits/sec CacheFlow 500:6 Mbits/sec CacheFlow 1000:15 Mbits/sec CacheFlow 2000:15 Mbits/sec	Active, automated, Prefetch/ Both/ URL filtering	HTTP, FTP/ ICP, CARP, WCCP/ up to 16
Cisco Systems www.cisco.com/ Cache Engine/ Appliance	1 10/100 Mbits/sec, Ethernet port	900	Not known/ 6 Mbits/sec	Passive/ Both/ User authentication, URL filtering	HTTP/ ICP, WCCP/ up to 32
IBM Corporation www.ibm.com/ Web Cache Manager / Appliance	Upto 4 10/100 Mbits/sec,FDDI or ATM ports	500	Not known/ 18 Mbits/sec	Active; scheduled prefetch./ Both/ User authentication, URL filtering	HTTP, FTP, DNS/ RCA, proprietary/ with E-Net dispatcher software , no theoretical limit.

IBM Corporation
www.ibm.com/
Websphere/
Appliance
	Software;
	Windows NT;
	Unix
		500	Not known/
			
			Not known
				Active; scheduled prefetch./
				Both/
				User authentication, URL filtering
					HTTP,FTP,
					DNS/
					RCA,
					proprietary/
					up to 32

InfoLibria www.infolibria.com/ DynaCache/ Appliance	2 10/100 Mbits/sec , Ethernet ports	64000	1000/ 45 Mbit/s	Passive/ Both/ User authentication, URL filtering	HTTP/ ICP, CARP, WCCP/ no theoretical limit.
Inktomi Corp. www.inktomi.com/ Trafficserver/ Appliance	Software Windows NT; Unix	10,000	1059/ 54.7 Mbits/sec	Passive/ Both/ None	HTTP, FTP, NNTP, RTSP/ ICP, WCCP/ no theoretical

					limit.
Microsoft. www.microsoft.com/proxy/ Microsoft Proxy Server 2.0/ Proxy Server	Software Windows NT;	Not known	Not known/ Not known	Active; scheduled prefetch/ Proxy/ Firewall; User Authentication; URL filtering; VPN encryption.	HTTP, FTP/ CARP/ no theoretical limit.
Netscape. www.netscape.com/ Netscape Proxy Server/ Proxy Server	Software Windows NT; Unix	Not known	Not known/ Not known	Active; scheduled prefetch./ Proxy/ User authentication, URL filtering	HTTP,FTP/ ICP , CARP/ no theoretical limit.
Network Appliance www.networkappliance.com/ NetCache C720 NetCache C720s NetCache C760/ Appliance	NetCache C720 and C720s:upto 6 10/100 Mbits/s. Ethernet ports with optional 1000-Mbits/sec Ethernet , FDDI ports. NetCache C760 : upto 8 10/100 Mbits/s Ethernet ports with optional 1000 Mbits/s , Ethernet , FDDI ports.	C720,C720s: 8000; C760:16000	1000/ 22.5 to 45 Mbits/s	Passive/ Both/ User authentication, URL filtering	HTTP , FTP/ ICP, WCCP, NetCache Clustering protocol (proprietary)/ no theoretical limit.
Network Appliance www.networkappliance.com/ NetCache Departmental Proxy, Netscape Enterprise/ISP/Proxy Server	Software;Unix	C720,C720s: 8000; C760:16000	400/ Not known	Passive/ Both/ User authentication, URL filtering	HTTP , FTP/ ICP, WCCP, NetCache Clustering protocol (proprietary)./ no theoretical limit.
Novell Inc www.novell.com/ BorderManager Suite ; fastCache/Proxy Server	Software;Netware	100,000	400/ 60 Mbits/sec	Active; scheduled prefetch./ Proxy/ User authentication, URL filtering	HTTP, FTP/ ICP/ no theoretical limit.
Entera Inc www.entera.com/ TeraNode Cache A-Series,E-Series, M-Series and R-Series/ Appliance	1 10/100 Mbits/s Ethernet port	Not known	A-Series:80 E-Series:128 M-Series:328 R-Series:328/ A-Series: 6 Mbits/s; E-,M-,R-Series: 45 Mbits/s	Passive/ Both/ Firewall,User authentication, URL filtering.	HTTP,FTP DNS,NNTP/ ICP/ no theoretical limit.

Table 1: Product Comparison

Back to Table of Contents

Summary:

Caching and replication is a growing market with a increasing number of products becoming available in the market. Caching and replication architectures are moving away from a hierarchical architecture towards a cluster based meshed architecture. ICP, WCCP and CARP are the most widely used protocols for caching. We also saw the use of Multicast based protocols to perform caching and replication. As mentioned earlier in the report, neither can be used in isolation.Increasingly, national caching and replication systems are being setup to reduce network latency in the backbone. Among caching products proxy server caches have a cost advantage over Appliances, even though the later display better performance. In the coming years caching and replication systems will be increasingly used to reduce latency and achieve better load balancing across the internet.

Back to Table of Contents

References:

The following references are organized approximately in the order of their usefulness and relevance.

[DraftComm]"Client Cache Communication", Internet Draft , Nov 98, 4 pages, ftp://ftp.nordu.net/internet-drafts/draft-melve-clientcache-com-00.txt
Describes Client Cache taxonomy and mentions all relevant protocols.

[DraftInterCache]"Inter Cache Communication Protocols", Internet Draft , 5 pages , ftp://ftp.nordu.net/internet-drafts/draft-melve-intercache-comproto-00.txt
Describes the prevalent Inter Cache Communication Protocols

[DraftHTCP]"Hyper Text Caching Protocol", Internet draft , August 1999 , 14 pages, ftp://ftp.nordu.net/internet-drafts/draft-vixie-htcp-proto-05.txt
Describes the Hyper Text Caching Protocol.

[DraftWCCP]"Web Cache Coordination Protocol", Internet Draft , June 1999 , 11 pages, ftp://ftp.nordu.net/internet-drafts/draft-ietf-wrec-web-pro-00.txt
Discusses the Web Cache Coordination Protocol

[DraftWPAD]"Web-Proxy Auto Discovery Protocol" , Internet Draft ,July 1999 , 18 pages, ftp://ftp.nordu.net/internet-drafts/draft-ietf-wrec-wpad-01.txt
Describes the Web-Proxy Auto Discovery Protocol.

[DraftTax]"Internet Web Replication and Caching Taxonomy" , Internet Draft , Oct 27 1999, 29 pages,
ftp://ftp.nordu.net/internet-drafts/draft-ietf-wrec-taxonomy-02.txt
A comprehensive list of all the current protocols in caching and replication.

[RFC2186]"Internet Cache Protocol (ICP), version 2" , Sep 1997 ,9 pages , ftp://ftp.nordu.net/rfc/rfc2186.txt
A detailed description of the Internet Cache Protocol (ICP).

[RFC2187]"Application of Internet Cache Protocol (ICP), version 2", Sep 1997 , 24 pages , ftp://ftp.nordu.net/rfc/rfc2187.txt

Discusses the cache hierarchies , protocol application , firewalls , Multicast and security consideration when using ICP.

[RFC2616]"Hypertext Transfer Protocol -- HTTP/1.1", June 1999 , 176 pages , ftp://ftp.nordu.net/rfc/rfc2616.txt
Describes HTTP 1.1 in detail .

[RFC1794]" DNS Support for Load Balancing", April1995 , 7 pages , ftp://ftp.nordu.net/rfc/rfc1794.txt
Describes DNS support for Load Balancing , including BIND and TTL algorithms.

[Wessels98] Duane Wessels and K Claffy,"ICP and the Squid Web Cache", IEEE Journal on Selected Areas in Communication, April 1998, Vol 16, #3, pages 345-357 , http://ircache.nlanr.net /~wessels/Papers/
ssDiscusses the functionality of the ICP and its implementation in Squid .

[Rousskov98]Alex Rousskov, Duanne Wessels ,"Cache Digests", April 98, http://wwwcache.ja.net/events/workshop/31/rousskov@nlanr.net.ps
Describes Cache Digests , including ideas behind the Cache Digest , its implementation in SQUID and a performance comparison with ICP in SQUID.

[Summary98]L.Fan, P. Cao, J. Almeida and A. Z. Broder," Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol ", SIGCOMM98 , http://www.cs.wisc.edu/%7ecao/papers /summarycache.html
Discusses the Summary Architecture and the concept of bloom filters used in this architecture.

[Sanjoy99]Sanjoy Paul, "The IPWorX Caching System", August 99, www.lucentipworx.com
Describes the IPWorX Caching System , a proprietary caching system from Lucent .

[CRISP98]Syam Gadde, Jeff Chase, "A Taste of CRISP", May 98 , http://www.cs.duke.edu/ari/cisi/crisp/crisp-wisp/
Gives an introduction to CRISP from Duke University.

[Touch98]Joe Touch, Amy S. Hughes, "The LSAM Proxy Cache - a Multicast Distributed Virtual Cache", August 98, http://www.isi.edu/touch/pubs/cnis98/
Gives an introduction to LSAM from the Information Sciences Institute at USC.

[CARP97]Microsoft Corporation,"Cache Array Routing Protocol and Microsoft Proxy Server 2.0", 1997, www.microsoft.com/proxy
Describes the Cache Array Routing Protocol and its implementation in the Microsoft Proxy Server 2.0.

[Tewari98]Tewari, Renu, Michael Dahlin, and Harrick Vin. "Design Considerations for Distributed Caching on the Internet." The University of Texas at Austin, Department of Computer Sciences, Technical Report TR-98-04, October 1998, 14 pages , http://www.cs.utexas.edu/users/UTCS/techreports/index/html/Abstracts.1998.html#TR-98-04
Describes the shortcomings of traditional caching systems and gives design criterion for designing distributed caches .

[Reardon99] Marguerite Reardon,"Caches Keep Content Close At Hand", Data Communications, March 21 1999, Pages 47 - 55.
Describes the current caching solutions available in the market and is a good source to find out product information.

[Netcache99]Peter Danzig and Karl L. Swartz , "Transparent , Scalable , Failsafe Web Caching", Technical report, Network Appliance www.netapp.com/technology/level3/3033.html.
A good source to understand L4 & L7 switching .

[Overview99]Internet Caching Resource Center,"An overview of Web Caching" , http://www.caching.com
A very elementary article which just introduces caching and replication.

[CISCO99]Cisco, "Network Caching and Cisco Network Caching Solutions", June 1999, http://www.cisco.com/warp/public/cc/cisco/mkt/scale/cache/tech/cds_wp.htm
Explains the Cisco Cache Engine series of products and describes WCCP.

Back to Table of Contents

List of Acronyms:

CARP	Cache Array routing Protocol.
DNS	Domain Name Server.
FDDI	Fiber Distributed Data Interface.
FTP	File Transfer Protocol.
HTTP	Hyper Text Transfer Protocol.
HTCP	Hyper Text Caching Protocol.
ICP	Internet Caching Protocol.
LSAM	Large Scale Active Middleware.
WCCP	Web Cache Coordination Protocol.
RCA	Remote Caching Array.
WPAD	Web Proxy Auto Discovery.
UDP	User Datagram Protocol.
RTSP	Real Time Streaming Protocol

Back to Table of Contents

Last Modified: November 18,1999.
Note: This paper is available on-line at http://www.cse.wustl.edu/~jain/cis788-99/web_caching/index.html