High Speed Content Delivery

Ronald Bhuleskar

rbhuleskar@scu.edu

Santa Clara University
Winter 2010

Audience:

The primary focus in this paper would be a network designer, designing network architecture for a high speed content delivery network with a primary focus of best effort data delivery with multimedia contents.

Table of Contents

Table of Figures

Part I

Introduction

1.1. What is content?

Lou Rousenfeld and Peter Morville defines content as "the stuffs in your website". Throughout this paper we will be focusing on the web content which consists of data, audio, video, image and animations.

1.2. Centralized Web

The traditional model of recognizing client-server architecture is the centralized web, where there is a single server with multiple users accessing it. All user traffic flows in the direction to the server creating lot of issues like traffic, single point of failure, etc. Some major drawback of not using the client server architecture in the business websites are as follows:

- Slow: Distance between Client and server may be huge, so the content travelling from client's location to server may take long time.
- Un-reliable content delivery: Congestion might prevent the delivery of the content.
- Not scalable: Having limited bandwidth at the server might lead to a non-scalable architecture.
- Poor streaming quality: Factors such as congestion & packet loss may degrade steam quality.

1.3. Content Delivery Networks (CDN)

With the rapid growth of Internet and services, the services competes each other on the limited network resources. High Responsiveness and Availability are keys factors for an effective website. Multiple user access the same website creating "Flash Crowd"[1] at a server. CDN emerged as a solution hence improving user satisfaction by eliminating the above listed issues. This is achieved by keeping data close to the user hence i) reducing the load at the server as each user access the nearest available server ii)Network traffic distribution & iii) Reduced latency

In the CDN there will still be an origin server which features replication of data to the other edge servers hence maintaining the consistency of data. The user needs to be properly routed to the edge server which will help achieve the CDN features. Let's get a glimpse of the actual use of CDN, considering a website www.yahoo.com which holds images on their website. Yahoo has lot of more important work to do rather than just transferring the same image again and again to multiple requests for the same website. It is very obvious to cache those images and bypassing delivery of those images by Yahoo. Hence Yahoo server will not be overloaded also low traffic at the yahoo server because lot of image transfer overhead is resolved by someone else. This just explains the

overview of how CDN works; we will go in much detail in the subsequent sections focusing on caching and load-balancing.

Scott Hull classifies different types of CDN based on the implementation while the concept of CDN and the services are the same for any type of CDN. Some of the common ones are Internet CDN, Overlay CDN, Peering CDN, Hosting CDN, Enterprise CDN, Hosting CDN [1]. Not going much into detail of their implementation types we will focus in general how CDN works.

1.4. Planning CDN Deployment

When it is decided to plant a CDN it is more important to locate proper position for the server. The server should be placed in such a position such that the delays [1] (transmission and computing) are reduced whether it be single-site single-server, single-site multiple-server, multiple-site single-server per site or multiple-sites single-server per site model [1]. The traffic pattern needs to be analyzed with tools like Performance Monitor to grab information like originating country & more. These web-log statistics needs to be properly analyzed by the network engineer before deployment.

1.5. Study of Akami

A company established in 1998 with the focus on effective content distribution and delivery network with more than 40,000 servers today is one of the largest CDN available today. Companies like Yahoo, and many business sites who wants their user to have high speed content delivery used Akami's network to store data. Akami then distributes the given data on their available servers across the globe depending on the requirement. The commercial website then asks commercial CDN like Akami to hand over content to their user achieving the benefits of CDN. This is how it works:

1) Yahoo signs up with Akami to provide content distribution to their users

2) Yahoo then provides all the contents (embedded objects) to Akami which needs to be delivered like audio, video, images, etc

3) Whenever user request Yahoo a webpage, it returns the code to the user with the information to retrieve the contents from Akami and other processed information.

4) User gets the information to be displayed from Yahoo and asks Akami to provide additional content. This reduces Yahoo's overhead to transfer heavy weight content to its user's.

Elaborating on the fourth point described above the main criteria is to find the optimum Akami server. Akami has not just few but thousands of servers to choose from. Depending on the location of the requesting client, DNS server chooses the appropriate akami server to fetch the content from.

Part II

Caching

Caching represents storing of information in a separate storage device to facilitate subsequent requests requesting the same information expediting the delivery from the cache. Cache can be a RAM or a disk or both. Several types of cache include Browser caches, proxy caches, gateway caches, server caches and application caches. Let's look at each one in detail. For any type of cache the function would be similar as described in the figure below:

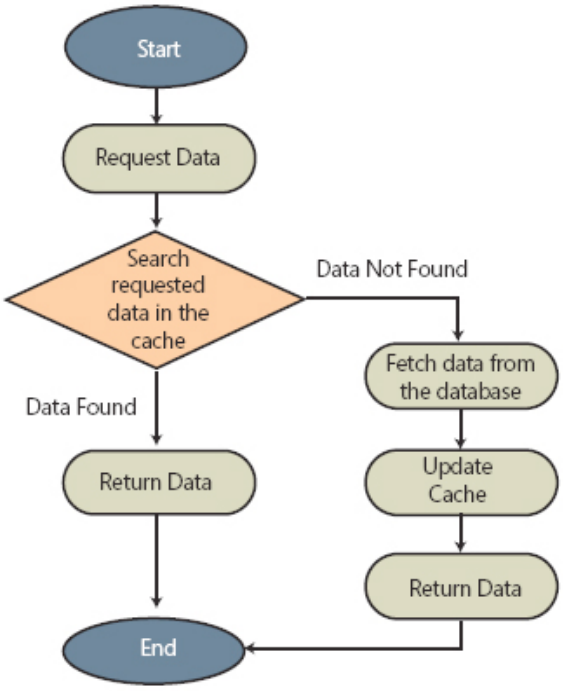

Figure 1: Flow diagram of retrieving contents from caches

2.1. Types of Caching

2.1.1. Browser caches

A simple caching technique is to use a browser cache where page once loaded by the browser it can serve the subsequent request to the same page from the browsers local cache.

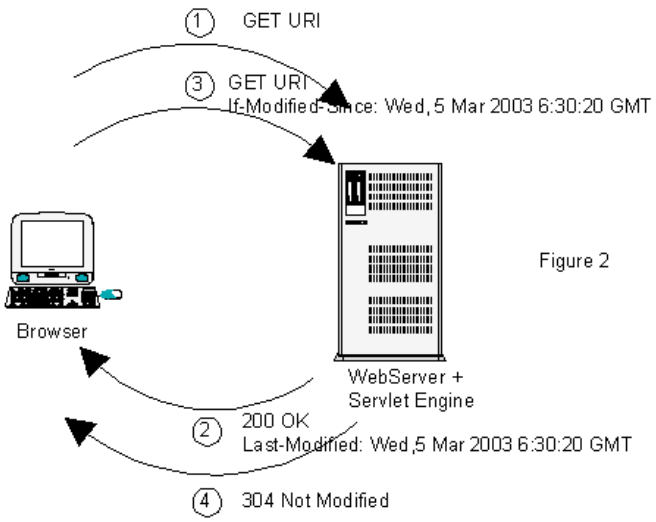

Figure 2: Browser Caches

The problem here is the consistency of data in the cache. Hence a duration can be defined till the cache is valid and after that no requests are served from the cache instead a new page is fetched from the server. This is achieved by having a validation check if the page is modified. Hence client browser requests for the page information rather than the contents itself. It then verifies the modification date with the modification date of the already existing page available in the cache. If it differs then a fresh copy of the page us fetched else the page already available in the cache is served.

The figure below gives a simple example of how browser cache is being used. The page default.aspx uses a javascript file topMenu.js and it is mandatory that before executing the file default.aspx you need to download topMenu.js. The browser first checks in its cache if the file was downloaded by some other page or by earlier visit to this page. If it is available then the browser can get the file from local cache itself hence saving bandwidth and time.

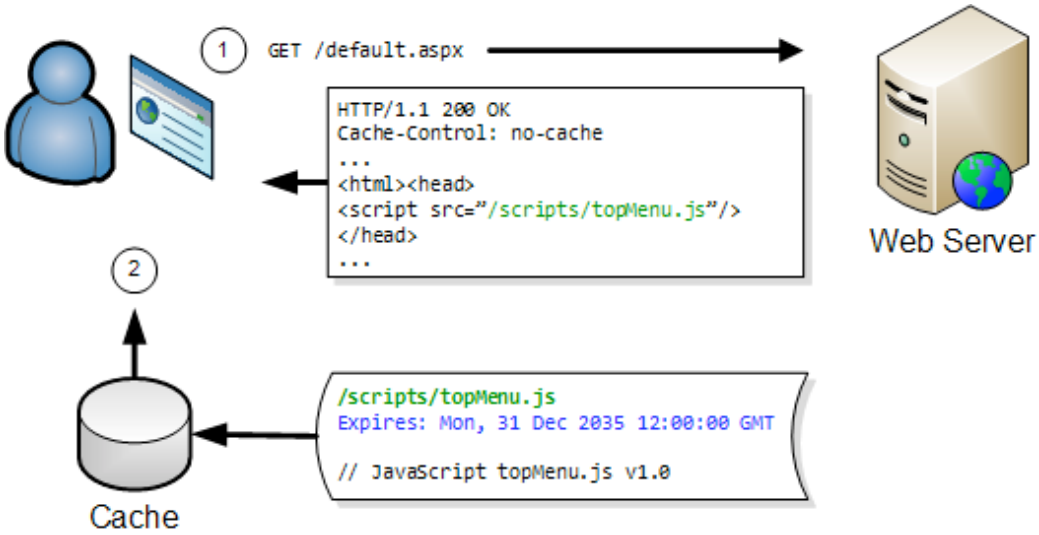

Figure 3: Browser caching demonstrated with client requesting webpage served by the cache

2.1.2. Proxy caches

A proxy cache resides on the edge of the network serving thousands of computers. The functionality is very similar to browser cache where pages are cached but can view as a shared browser cache where pages from multiple users are cached. The users' requests are routed through the proxy to see if the page is available with the proxy.

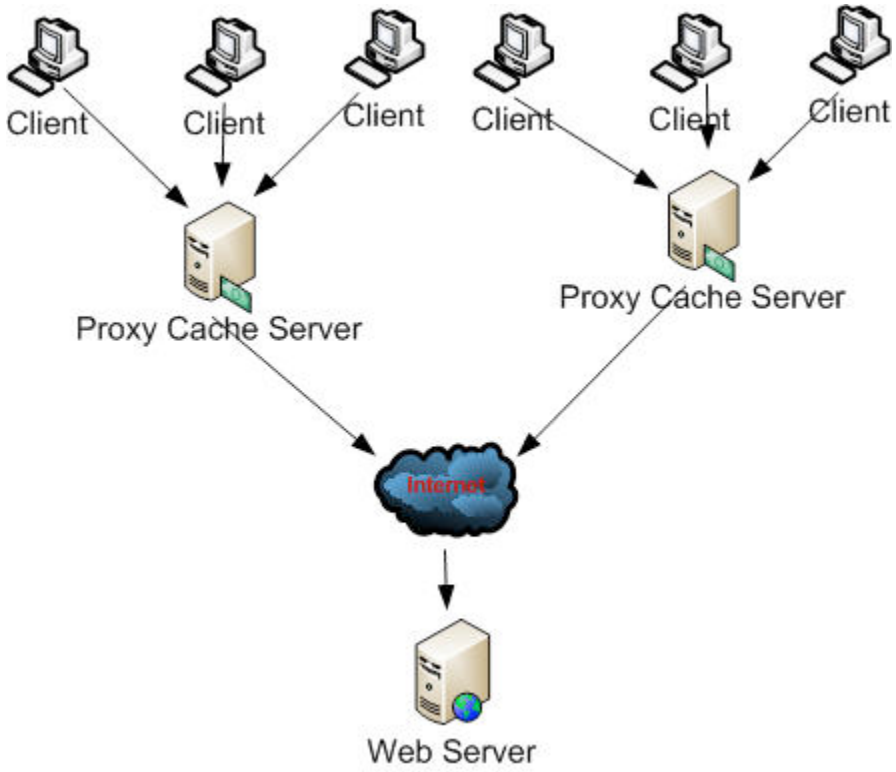

Figure 4: Proxy caches

The advantage of this is if user A requests a webpage which wasn't available in cache but a request was made to the server to get the page. Subsequent requests from other user on the network will be served from cache itself. These could be hardware devices built with the firewalls or router. This would help to deliver contents to the end user more efficiently speeding up the delivery.

The best placement of the proxy cache is at the ISP site hence increasing the content delivery rate for its customers increasing customer satisfaction. The above figure describes a scenario where multiple clients are connected with the ISP where a proxy cache is installed and which provides connectivity to the Internet. This is also known as forward-caching.

2.1.3. Gateway caches

Gateway cache is installed by the sites' web-master to provide high speed content delivery to its site users. It can be thought of a reverse proxy cache which resides in front of the web-server used by distributed browsers whose request can originate from different

network. These devices can be distributed to other locations on the Internet achieving the capability of a CDN (Content Delivery Network). This is also known as reverse-caching.

2.1.4. Server caches

Operating System provides a functionality of caching data and storing it in the memory. The application server providing web-server capabilities may allow frequently used webpage to be cached in the main memory and served faster. For e.g. a homepage of the website which is requested very frequently could be cached in the RAM itself.

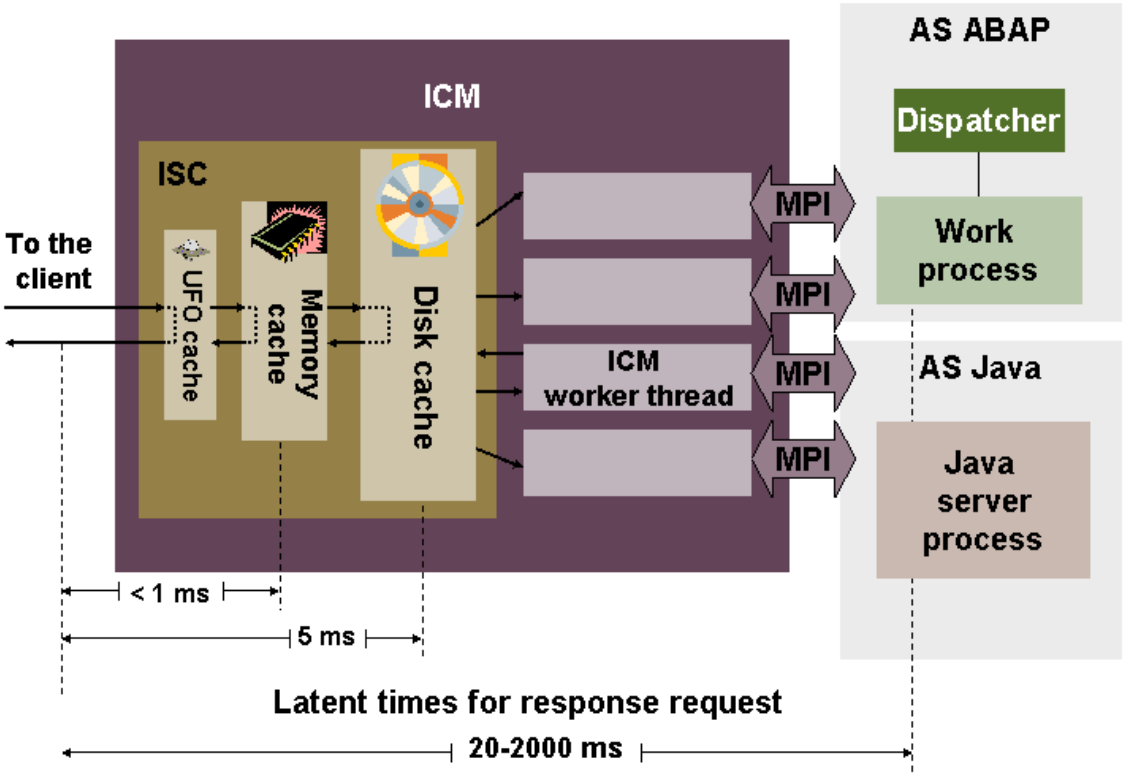

Figure 5: Latent times for response request served at various levels of caches

From the above figure not going much into the detail, it could be clearly seen that the time taken for the request to be served if the page is served from RAM is less than 1 milliseconds, while on Disk cache it takes nearly about 5 milliseconds and when the page is not cached, processing the page takes about 20 milliseconds to up to 2 seconds.

2.1.5. Application caches

The programming language constructs to define how a page can be cached. ASP.NET defines rules which allow page caching. The rules need to be declared with the file to be cached. A simple example of the output cache directive in ASP.NET is as follows:

<% Output Cache
    Duration = “seconds”
    Location = ”Any|Client|Downstream|Server|None”

Vary By Custom = “browser|customstring”
Vary By Header = “headers”
Vary By Param = “parametername” %>

The attribute duration allows till how much duration a cache copy could be used without recompiling the byte-code. Till the valid duration the same cache copy could be served. Another attribute Location allows placing the cached copy of the webpage. The value Any can keep the cache copy of the webpage either with the browser, on webserver or on any proxy between client and server supporting HTTP1.1 caching. The value downstream indicates storing cache contents anywhere other than the client’s browser. To cache different versions of the same page Vary by option can be used.

2.2. HTTP headers for caching contents

HTTP1.1. Protocol has currently more than 50 HTTP headers [2] which are subdivided into entity, general, request and response. Entity header is used to contain information about body or resource while general header can be used for both request and response headers. Request header is used to send information from client to server while the opposite response sends information from server to client. Request usually asks for a web-page with few parameters required for processing at the server while Response gets the processed http page to the client to display. Following header is a typical HTTP Response header.

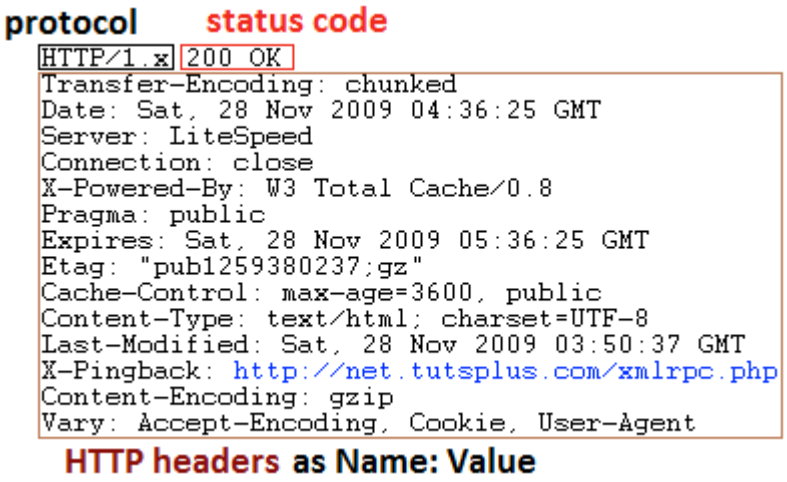

Figure 6: HTTP Response Header

The date field identifies the time at which the http response was originated from the server. It has many other parameters like Server, Connection, Pragma, etc. Not going into details in all of them we will see only those which are useful to us. You can find more in detail about HTTP Response header in [1] & [2].

Expires header: It informs all the caches the expiry of data, after which the data might get stale. It tells the nodes having cached the data not to serve users with the cache data once the time pass the Expires time. Instead a fresh copy could be requested from the source and cached again.

Cache-control header: A new feature in HTTP 1.1 which defines how the caching should be performed on the request/response path. Cache-control can be divided into request and response headers. Each sub-header can have multiple directives under it defining a unidirectional caching scheme. It is very helpful for website that needs more control over their contents.

Header Format:

cache-control : "cache-control" cache-directive : "cache-request-directive" | "cache-response-directive" "cache-request-directive" | "cache-response-directive" = "no-cache" | "no-store" | "max-age" = seconds | "max-stale" = seconds | "min-fresh" | "no-trasform" | "only-if-cached" | cache-extension

From the Figure 6. cache-control : max-age=3600 means the contents would be cached for 3600 seconds i.e. 1 hour. Detailed information about the cache-control directives could be found in [1].

2.3. Transparent vs. nontransparent caching

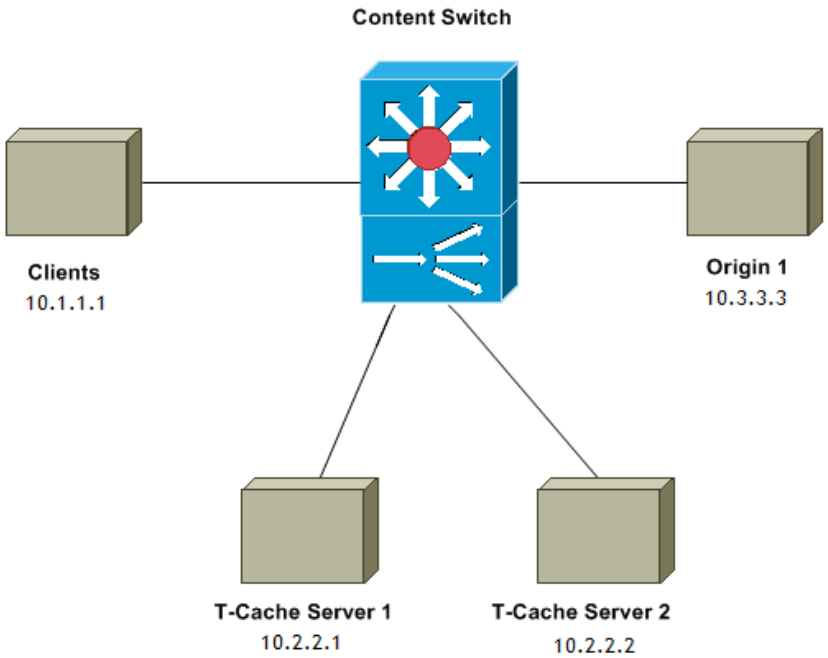

Figure 7: Caching data in network demonstrating transparent and nontransparent caching.

In nontransparent caching the client (10.1.1.1) address the proxy cache either 10.2.2.1 or 10.2.2.2 as a destination IP. If the proxy cache has the content then it is returned back to the client else it's the proxy's responsibility to create a new TCP connection with the server (10.3.3.3) who actually has the content to be delivered.

In transparent caching the Client directly address the destination 10.3.3.3.The proxy is configured such that all traffic goes through it. The client believes that he would receive the response back from the Server (Origin1) but it could be delivered either with the proxy server (T-cache-server-1 or T-cache-server-2). On a cache-miss the proxy will obtain the contents from the server to serve the client.

2.4. "Akamaizing Content"

The work "akamaizing" has become so popular in IT industry that became a synonym that the contents are automatically CDN aware. Typically web-sites need to go through a time consuming process for the contents to be CDN-aware. The web sites also update their web application that they generate a CDN-aware output. There are automatic tools available by Akami and Speedera but still it remains a time consuming process. Reverse-proxy-caching would be easier to cache contents dynamically without any additional work. The contents are then automatically re-written for any CDN. Hence it reduces much load on the Origin servers.

An example to see how contents are written by 2 major CDN providers, Akami and Speedera.

A webpage www.scu.edu without a CDN may look like this:

```
<HTML>
<CENTER> SANTA CLARA ENGINEERING </CENTER>
<IMG SRC="pic1.jpg">
<IMG SRC="pic2.jpg">
</HTML>
```

When contents are re-written by Akami the contents would look like this:

```
<HTML>
<CENTER> SANTA CLARA ENGINEERING </CENTER>
<IMG SRC="http://a7.g.akami.net/5/6/21/2222/www.scu.edu/pic1.jpg">
<IMG SRC=" http://a7.g.akami.net/5/6/21/2222/www.scu.edu/pic2.jpg">
</HTML>
```

When contents are re-written by Speedera, the speedera-ized site would look like this:

```
<HTML>
<CENTER> SANTA CLARA ENGINEERING </CENTER>
<IMG SRC="http://cdn.scu.com/pic1.jpg">
<IMG SRC=" http://cdn.scu.com/pic2.jpg">
</HTML>
```

Each CDN company uses its own static prefix string for the pic1.jpg and pic2.jpg but the main principle of any CDN remains same, the client DNS query against the CDN DNS server to make it point to the closest possible cache server where contents are available.

2.5. Using ESI: Edge Side Includes

Akami believes that more requests are services and the pages are assembled closer to the user. But some contents which are not static needs to be taken from the application server running at the Origin (Server who hosts the contents) or contents might change every specific period of time. Hence Akami in collaboration with content management and application server organization came up with the idea of ESI: Edge side includes. ESI is a markup language very much like an HTML used to define contents to be dynamically assembled at the edge. Each file has its own TTL (Time to live) field which determines how much time a file can be cached after which the updated contents should be cached. This enables only small amount of contents to be fetched from the Origin and most of the data to be assembled from the contents available at the edge. For e.g.: Cookies could be retrieved from the Origin while the static contents could be retrieved from the edge. ESI helps to breakdown the contents into fragments hence inducing an advantage of separating the cacheable and non-cacheable contents. Hence, small fragment of code needs to be requested from the Origin. This could used when application server and edge server both collaboratively supports ESI.

2.6. CDN DNS

From the previous discussion it is clear that each CDN company puts a static prefix string to uniquely identify the content. These are only done to contents like audio, video, images because of their static nature. When a client wants to fetch the static content pointing to a CDN cache such as Akami's cache, a DNS query is made to a DNS server. It is obvious that the DNS server cannot blindly return the IP address because if it does then there will be no use of using a CDN for content delivery. The CDN DNS would instead find a closest CDN cache server/cluster where data would be available. Hence, reducing the latency of content being travelling a long distance between client and server. It is assumed that the cache clusters are located very near to the client's physical location. With Akami's CDN from 15,000 cache servers in 2003[2]; 20,000 servers in 2005[1]; 40,000 servers in 2008 and today having more than 45,000 servers in more than 70 countries across more than 1100 networks [3] boosting the performances of web-sites resulting into repeated visits of users.

Part III

Load Balancing

We have seen earlier the disadvantages of using single server approach for a business website. Another approach proposed was to use a Server farm. But still there are many issues which arise with the use of server farm listed below [2]:

- Traffic redirection to appropriate server in server farm
- Equal Traffic distribution across servers
- Traffic Distribution without overloading servers
- Resolving techniques if server is overloaded

3.1. Server Load Balancing

The primary focus of a server load balancer is to eliminate the issues listed above. Simple approach of load balancing is to deploy a load balancer that provide infrastructure to scale applications. The functions of these devices are to redirect traffic to caches, load balancing traffic across multiple firewalls, packet filtering and packet management [2]. These sophisticated devices uses OSI layer 3, 4 and 7 to identify application layer sessions which are helpful for re-direction. Scott Hull describes some ways the load balancer could distribute traffic.

- URLs for HTTP traffic
- UDP ports for DNS load balancing
- TCP ports for load balancing SMTP
- IP destination address for default gateway load balancing.

One of the load balancer pioneers, Alteon Web System, developer of the new more flexible load balancer named its devices as “Web Switch” also known as Content smart switch which works at layer 4 to 7.

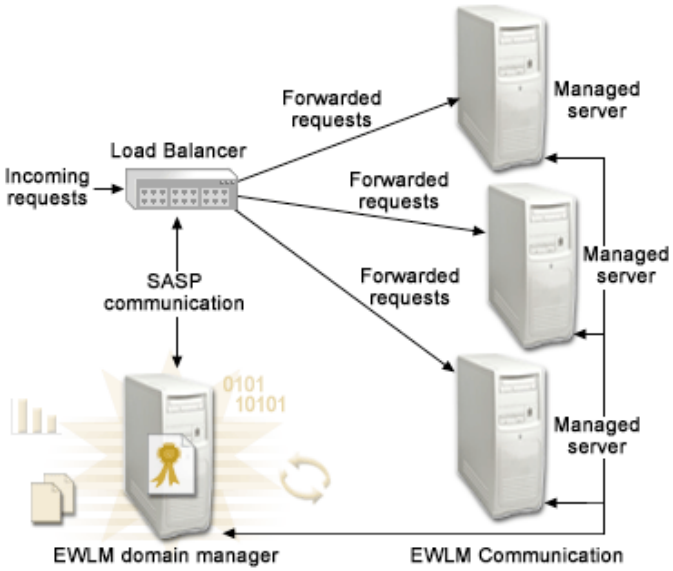

Figure 8: Load Balancer

The above figure gives a generalized view of a function of a load balancer. User Traffic flows into the load balancer. Load balancer depending on a function decides where the user traffic needs to be directed considering the issues described in section [2]. This could be implemented in software, hardware or combination of hardware and software. In the next section we will study different load balancing techniques.

3.2. DNS Load Balancing

This requires a site to have two or more servers running same software with their addresses associated with one domain name. The DNS issues these address in a round-robin fashion, providing equal number of hits to each server or load-sharing featuring equal amount of work at each server. With round-robin the client requesting address from the DNS receives the address of server 1, another client requesting address from the DNS receives the address of server 2. There are two methods proposed by Gilbert Held of DNS based load sharing:

- CNAMES
- A records

3.2.1. Using CNAMES

The most common implementation of DNS is the Berkley Internet Name Domain (BIND). Load Balancing under BIND4 is provides using CNAMES. Let's see how it is achieved. Consider an organization having 4 web servers with IP addresses 154.25.63.1, 154.25.63.2, 154.25.63.3 and 154.25.63.4. The CNAMES entry in the DNS would be as follows for the domain name www.cnames.com

```
Server1          IN      154.25.63.1
Server2          IN      154.25.63.2
Server3          IN      154.25.63.3
Server4          IN      154.25.63.4
```

You can now set the web server name (Server1-4) to any name but should match the canonical name to resolve www.cnames.com.

```
www   IN      CNAME           Server1.cnames.com
      IN      CNAME           Server2.cnames.com
      IN      CNAME           Server3.cnames.com
      IN      CNAME           Server4.cnames.com
```

Based on the preceding the DNS will resolve the www.cnames.com to any of the 4 servers (Server1-4) in the group in a rotational fashion distributing the load among the 4 servers.

3.2.2. Using A records

CNAMES only works for BIND4, For BIND9 and above multiple CNAMES cannot be used for recording a domain. Load balancing via A record DNS could be used as stated below:

```
www.arecords.com    IN      A       154.25.63.1
www.arecords.com    IN      A       154.25.63.2
www.arecords.com    IN      A       154.25.63.3
www.arecords.com    IN      A       154.25.63.4
```

A variation to the above is to have a Time to Live (TTL) field with A records. It indicates the amount of time after which it should be refreshed, which would be useful to maintain consistency of DNS cache. Following describes how a TTL could be used with A records:

```
www.arecords.com    60      IN      A       154.25.63.1
www.arecords.com    60      IN      A       154.25.63.2
www.arecords.com    60      IN      A       154.25.63.3
www.arecords.com    60      IN      A       154.25.63.4
```

In the above case the DNS cache server would be refreshed every 60 seconds. In a case of one of the server runs down there is no way the DNS can know that there should no traffic directed to that server. Hence, in a server farm of 4 servers only 3 would be active. Hence a round-robin DNS would serve 25% of wrong address to the overall client's requests.

3.3. Internet Protocol Address Mapping

Many a times when a server in not operational and has an entry in the DNS, it will still serve those addresses to the user resulting into an invalid address. This is because the DNS does not do any check to verify if the server is in operational state. Hence it happens that for some rare hit you don't get the result because the server might be un-operational. And a re-hit fetches you a page. This is because the content fetched from this address would be different from what you had got earlier.

Another load balancing technique maps a domain name with a single IP address. That machine is dedicated to perform load balancing that intercepts HTTP requests and distributes to multiple web server. This could be performed either with software or with hardware. The main feature of using this technique is that the load balancer machine could periodically check of the web servers are operational and can intelligently forward or re-direct request to them. This cannot be obtained with DNS load balancing technique.

3.4. Virtual Internet Protocol Addressing

A much smarter way of load scheduling is to configure multiple servers under a single virtual address using the technique of virtual IP addressing. With this technique many servers can be used to respond for one IP address. Considering the web server's IP address used in previous example 154.25.63.1, 154.25.63.2, 154.25.63.3 and 154.25.63.4 could be logically merged into a virtual IP address of 154.25.63.5. You can configure a scheduler (for e.g.: 154.25.63.1) which would be any one of the existing web server which then redirects the traffic to the other 3 available web servers. In this case if any of the web servers fails scheduler who actually is the web server can be notified or it can verify the mal-functionality of the web server and can intelligently stop traffic redirection to that server. In case if the server who performs load scheduling goes down another web server from the farm can stand up as the acting scheduler still remaining the IP 154.25.63.5 valid but just the scheduler which was initially 154.25.63.1 is now any one from 154.25.63.2, 154.25.63.3 or 154.25.63.4.

3.5. Load Balancing Methods

There are two possible implementation of load balancing scheme which should be considered before selecting a load balancing platform. One is a hardware based load balancing and other is software based. Each of them has their own advantages and disadvantages. Software based load balancing is much cheaper costing negligible as compared to a hardware based load balancing. Although hardware based load balancing provides higher serving capability it is not much used today.

The primary feature of a load balancer is to distribute load equally across various servers and to validate the existence of the server in the system. There are three popular load balancing method which distributes the HTTP traffic to various servers:

- Random Allocation
- Round-Robin Allocation
- Weighted Round-Robin Allocation

3.5.1. Random Allocation

Each incoming HTTP request is directed to the server in a random fashion. Though it is easy to implement, the disadvantage is one server may have more requests to be server than the other server.

3.5.2. Round-Robin Allocation

In this method of load allocation the load is distributed among multiple servers. The first http request is directed to the server1, second http request to server2 and so on. This continues in a circular manner, hence at any given point in time all servers will have exactly same load or one more than the load to the other servers in the group. The disadvantage of round-robin allocation is that it does not take into account the capacity of the server. A server with double the capacity of another still receives the same amount of load to be served as the one with half the capacity, hence, not utilizing the computing system efficiently.

3.5.3. Weighted Round-Robin Allocation

The drawback of round-robin allocation is eliminated by weighted round-robin allocation which allocates the requests according to the capacity of each server. This means that the high capacity server would be required to serve higher number of requests as compared to the lower capacity server, hence, utilizing the servers' computing resources efficiently. In reality this method of request allocation is not full-proof because it just redirects the amount of load depending on the computing power regardless of the type of the request. It is believed that not all request takes the same time to process. Hence, a load balancer should maintain a database to determine type of request and the processing time required by each server, accordingly, classifying the traffic with traffic parameters like request type.

3.6. Switch Processing

There are many companies like Avaya, A10 Networks, Brocade, Cisco Systems, Citrix Systems, Nortel Networks, etc who develops load balancers. Each has their own features some of them are described in detail in [2]. Any load balancer should be capable of handling thousands of incoming requests per second hence requiring a large memory and processing power. The three main processing type at any server load-balancer is: background processing, per-session processing and per-frame processing.

3.6.1. Background Processing

General server functionality such as doing health checks, generating statistical reports and topological communications. This is very light weight processing as compared to other processing done by the switch hence could be single threaded and can be handles by a co-processor inside a switch.

3.6.2. Per-session processing

The most important operation of the load-balancer switch is to determine the user session and accordingly directs the traffic. Considering the user request which was first directed to one server where some session information is stored. The subsequent request from the same user if re-directed to another user now creates problem because the session information is not available with the server where the user traffic is now re-directed. Hence the device should have some mechanism to determine where the user request was first re-directed and should re-direct the subsequent requests from the same user to the same web server maintain user sessions. The drawback is that in case of the server failure the entire session information kept on that server is lost. A solution for that is to create a database where session information is stored and all web server accesses this database to. Again a problem of central database is the central point of failure. Sometimes the database is even replicated at multiple locations increasing reliability so that all servers in the farm can use the same session information and the load balancer can re-direct users request to any server regardless where its previous requests were forwarded to. Using browser cookie and URL re-writing is another option [4] but fails for many with complex business logic structures.

3.6.3. Per-frame processing.

It includes packet parameters substitution like IP, MAC, Port number, or may be examining actual data. There will be thousands of frames entering the switch per second and hence it needs a very fast switch to process at such a rate. In reality for a high end system the centralized switch for load balancing may fail to re-direct frames and hence can be distributed to multiple servers on a data path.

Part IV

Compression

Compression is the technique to reduce the weight of the overall content to be transported over the network. Compression does not expedite data delivery but it is believed that reducing the contents to be transferred over the network can in turn expedite data delivery. There compression algorithm can be broadly classified as lossless and lossy compression, each having their own merits and de-merits and should be chosen depending on the application. A good compression algorithm will improve the content delivery drastically. This technique being out of scope of this paper we will not study it in detail. Much information could be obtained in [5]

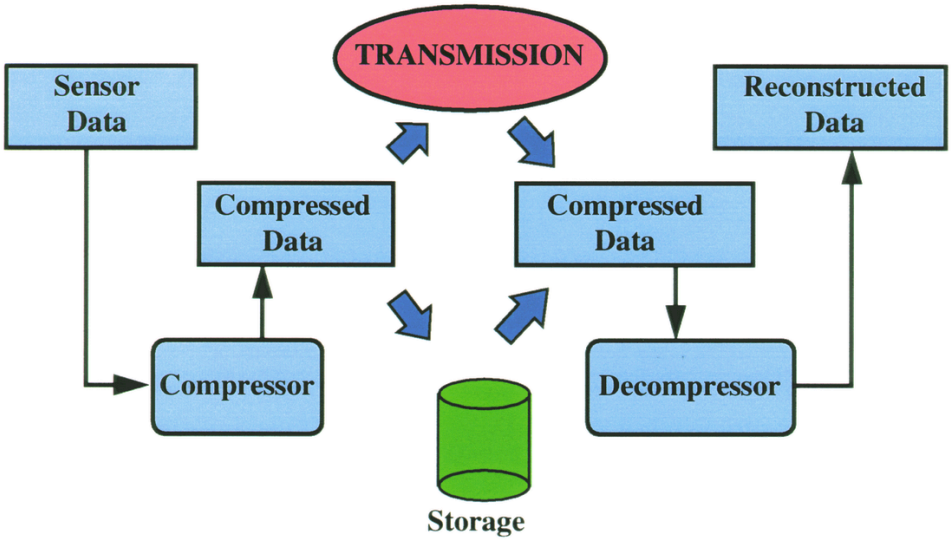

Figure 9: Data Compression module

The figure above describes when the data is to be compressed and de-compressed. Before the data is to be transmitted the data is given to a compression module and the output of the compressor is transmitted. The compressed data received by the receiver is then given to a decompression module to acquire the original data.

Part V

QoS

QoS is very important when it comes to transfer high quality videos or serving to a large volume of requests. For Video streaming QoS means that the content should be received at the receiver end without interruption, without quality degradation like frame loss. One of the most important parameter in serving a video to the end server with good quality of service is to have a buffer capable enough to store frames. Having average re-buffering rate parameter to determine the ratio of the time used to fill the buffer with the play out time. It is said that a high quality video streaming should keep the rate of re-buffering below 0.5% discarding the initial buffer, at normal bandwidth availability [6].

Another key parameter is the average frame rate and the rate of frame loss, these parameters gives an idea about the capacity of the server and the user receiving and deciding the video. The best user experience is when having a constant frame rate. In case of server overloading the option is to reduce the frame rate increasing the frame loss. Also when client does not have the full strength to stand capable of receiving and decoding the contents of the video these parameters could be tweaked.
With Flash Media Server, some parameters with regards to memory usage for input output buffer, size of buckets, and use of aggregated messages can directly influence the video frame rate.

5.1. QoS parameters for Video Streaming application

Learning some very important QoS parameters required for Video Streaming, let's just also see few basic parameters lightly affecting QoS. Connection establishment time, connection rate, sync between audio and server, Peak signal-to-noise ratio (PSNR) between video decoded by video and video that comes out of encoder. A good video streaming should consider all the QoS parameters described here. More details about the QoS could be obtained in [5].

Increased load of contents like file transfer, image, audio and videos increased load on the network. One option is to combine dedicated circuit with ATM technology. Today, audio and video are a part of a business website increasing load. Hence the TCP over the packet based network is responsible to solve their issues. The multimedia data are usually large, and creates bursty traffic when send across IP network. In the absence of ATM architecture, TCP/IP is responsible to accommodate bursty and stream traffic within TCP/IP architecture. Some techniques have been proposed in [7] to solve these issues.
There are some protocols used for QoS support like RSVP, MPLS, RTP, etc which are described in detail in [5] & [7].

Part VI

Quadruple Play

Till now we have seen the how contents can be served to the user efficiently providing services like voice, video and data called as triple play providing three services to their customers. Companies such as AT&T, Comcast, Verizon already provides such features to their customers offering triple play. Recent development or rather to say advancements in the triple way is the Quadruple play [8]. As the name suggest there is definitely a fourth feature added on to the triple play, its 'mobility'. The paper till now tells you how to achieve triple play services. Wireless and wired networks usually differs at the layer 1 and 2, and could be worked without modification at the layers 3 or above.

Achieving "mobility" in quadruple play.

Not going much in details about quadruple play we will see some important improvising features which were proposed at the layer 2 in following areas of the existing wireless infrastructure to achieve mobility in quadruple play.

- Packet scheduling
- Retransmission scheduling
- QoS guarantee and provisioning

6.1. Packet scheduling

Proper packet scheduling can effect achieve substantial performance in channel utilization, average packet delay and access energy cost [9]. One way to achieve this is to reduce the MAC overheads and improve the channel utilization. One way to achieve it is through pipelined packet scheduling.

The basic way of accessing media among multiple users is DCF (Distributed Coordinated function). DCF relies on CSMA/CA or RTS/CTS to share medium. 802.11 use an exponentially back-off algorithm to resolve channel contention. We briefly describe IEEE 802.11 DCF, for more details you can refer [9] and [10].

A station who wants to access a channel generates a random backoff value between [0, CW] where CW stands for contention window, it sits idle slot for which the station waits and later tries to access the channel. During the backoff procedure the backoff counter is decremented by 1 after each idle slot and freezes when channel is sensed busy. If a collision occurs the station's contention window is doubled unless its less than the CW_{max}. On a successful transmission the CW value is reset to the minimum value CW_{min} . In RTS/CTS access method an RTS is send to reserve the channel while CTS comes in confirmation to it. On a successful handshake a channel is granted for specific duration for which it was defined in the handshake procedure. All stations covertly listens to these messages and one who wants to access the channel hears this message suspend the transmission over the channel to a later time. This is called as "virtual channel sensing"

implemented using “Network Allocation Vector” (NAV). Each station updates its NAV according to the time mentioned in the MAC header packet it hears.

The figure below shows a RTS/CTS access method of 802.11 DCF. When a channel is sensed idle for duration of DIFS (DCF Interframe Space), each backlogged station does the backoff procedure. An SIFS (Shorter Interframe Space) is used to separate single dialog transmission for e.g.: CTS, Data and ACK frames in the case of RTS/CTS access method.

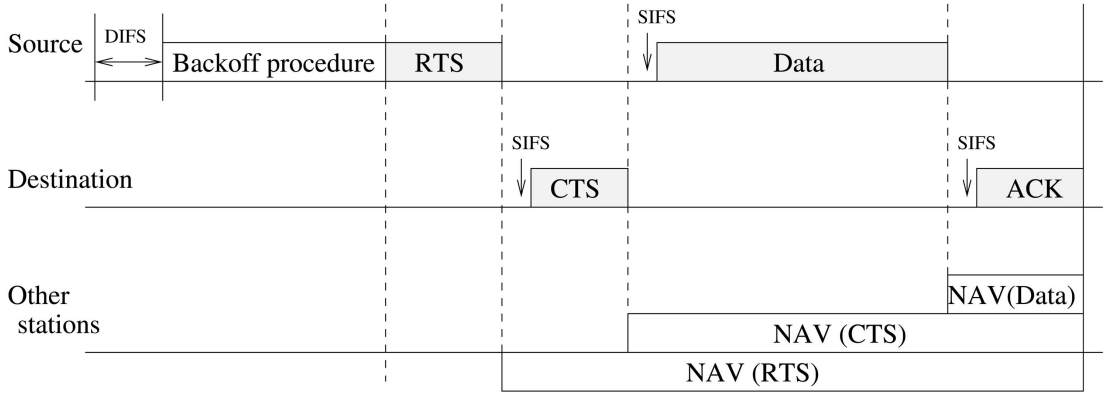

Figure 10: RTS/CTS access method in 802.11 DCF

6.1.1. Serialized Packet Scheduling

In IEEE 802.11 the channel first contends for a channel, acquires channel and then transmits the packet over the channel. This is the sequence of operations as shown in the figure below. This is called as sequential packet scheduling as contention resolution and packet transmission are serialized. In this case during the contention resolution, entire bandwidth is wasted.

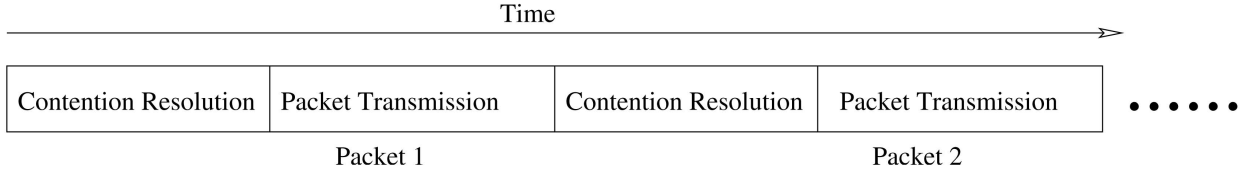

Figure 11: Serialized packet scheduling

A better approach for packet scheduling is pipelining the contention resolution in a different channel. Hence the channel can content for a given time duration on a separate channel while another channel transmits the packet. Hence after completion of transmission of the first packet it does not waits to resolve the stations to contend and decide which one will transmits the next packet instead it is already decided during the transmission of the earlier packet. Hence it can utilize the bandwidth efficiently. An example of 2 staged pipeline is shown below.

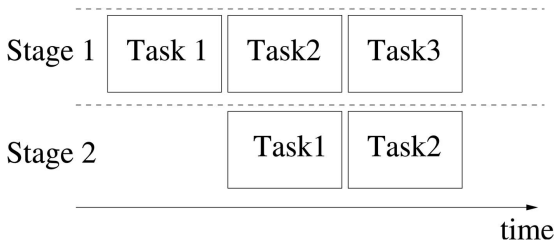

Figure 12: Example of 2 staged pipeline.

Medium access control job can be split into different stages which are scheduled on different subchannels. These multiple stages are processed in parallel such that overall channel utilization is improved. The channel can be split using time division multiple access (TDMA) or with frequency division multiple access (FDMA).

There are many schemes to handle two-phased pipelined packet scheduling in 802.11 DCF viz., "Dual Channel Pipelined Scheduling" (DCPS), "One-Phase Busy Tone Pipelined Scheduling" (One-Phase BTPS), "Two-Phase Busy Tone Pipelined Scheduling" (Two-Phase BTPS). These could be found in more detail in [9]

6.2. Retransmission Scheduling

Channel errors and packet losses are frequently seen in wireless network. Usually the packet loss is masked through the link level transmission by retransmitting the packet until it is transmitted successfully (Maximum of 4 retries). Link level re-transmission in Wireless LAN's has 2 major drawbacks [11]. First, the packets sent over unstable links are more likely to be lost than the unstable links, and sending more packets to the unstable link causes overall performance degradation. Second, repeated retransmission of other packets blocks other packets and causes delay, which could be very fatal for a delay-sensitive traffic.

There has been lot of efforts taken to retransmit the lost packet on a unstable links. Packet Fair Queuing Algorithm is proposed in [12] to guarantee long term resource fairness in wireless networks. It helps improve overall throughput but still the problem of immediate retransmission leads to delay of the other packets.

Sungwon Han and Ikjun Yeom proposed a link level retransmission scheduling algorithm to eliminate the additional delay needed to retransmit the packet immediately. When the packet sent over the link is lost we consider the node connected on an unstable link. Rather than retransmitting the packet immediately, packets the kept in the retransmission queue, which itself schedules the packet contained in the queue to be transmitted. Whenever the node receives a chance to transmit its content the packet from the retransmit queue is transmitted first. As soon as stable link appears to the node the retransmit scheduling should be stopped. Here the impact of transmission on delay of other packets is removed since the retransmission occurs only on the permitted time. After implementing the proposed idea in ns2 the author concluded that overall throughput increases through reducing bandwidth wastage due to keeping sending packets over unstable links which are likely to be lost [11].

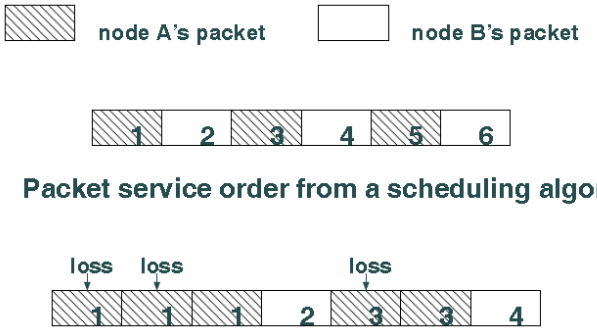

Figure 13: Packet transmission without re-transmission scheduling

Figure 14: Packet transmission with re-transmission scheduling

The above figure is self explanatory, it tells that the node transmits packet only when it gets it timeshare to transmit. Unlike immediately transmitting the packet as shown in Fig a) it keeps in the transmission queue unless the channel is acquired till then it waits in the re-transmit queue as shown in figure below. The node has to exit from the retransmit scheduling state once it knows that the link is stable. The way it is done is by incrementing a counter each time a successful retransmission takes place and decremented for each packet loss. After a certain threshold the node learns about a stable link and can now send the packet/s available in the retransmission queue and resets the count.

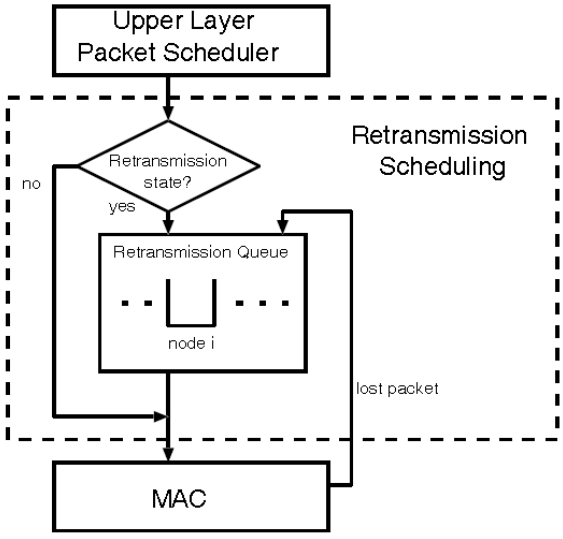

Figure 15: System architecture for retransmission scheduling

6.3. QoS Significance

MAC protocols are very successful and robust for best effort traffic but it is unsuitable for multimedia content delivery with the QoS requirements. A node might wait for an indefinite period waiting to transmit a frame hence a QoS for such traffic is very important. Centrally controlled MAC protocols and reservation based protocols are much better but are hardly implemented in today's devices because of its complexity and inefficient for normal data transmission [7]. There was a "war" between the Asynchronous Transfer Mode (ATM) and IP network years ago as which one provides better QoS support for data transmission. QoS support in ATM network is relatively easier than in IP but ATM failed for many reasons. Finally IP won the war because of IP QoS Protocols such as Integrated Services (IntServ), Differentiated Services (DiffServ), Multiprotocol Label Switching (MPLS). We should provide QoS solutions to the end users for both contention based and non contention based MAC protocols. Paper [7] describes the work in QOS guaranteeing and provisioning at the contention based Wireless MAC layer. Without a QoS support at the MAC layer QoS support ant any higher layer is impossible.

IEEE 802.11 WLAN is mainly popular because of the DCF which we had studies in earlier part of this paper, PCF (Point Coordination Function) is hardly implemented today because of its complexity. Research has proved that, DCF itself stands incapable to handle multimedia data transmission with QoS requirements. Voice and Video traffic are time sensitive and frame delays are unacceptable. One solution defined in [7] is to modify DCF for a priority scheme. DCF priority scheme can easily be designed with minor changes in existing DCF, which would be quite effective. IEEE 802.11e MAC employs a channel access function, the hybrid channel function (HCF), which includes contention based and centrally controlled channel access, which is also called as enhanced distribution coordination function (EDCA). EDCA provides a priority scheme by differentiating the interframe space as well as the initial (CW_{init}) and maximum window (CW_{max}) sizes for backoff procedures [7].

6.3.1. QoS Guarantee

A distributed admission control is for differentiation services of the EDCA. Channel utilization measurement is conducted during each beacon interval and available/residual budget are calculated. When once class budget gets reduced, new traffic streams belonging to this class cannot gain transmission time and the existing nodes will not be allowed to increase their transmitting time. Therefore the existing traffic streams are protected, utilizing the channel capacity.

6.3.2. QoS Protection

Much channel capacity can be utilized by existing voice and video, too many best effort data transmission can affect the voice and video flows since many data transmission can cause collisions. Hence voice and video flows becomes vulnerable to data transmission. The reason behind that is that the priority support is not strict. Proposed data control mechanism in [7] helps in protecting the voice/video flows from best effort data traffic.
It would be unfair to limit the number of stations accessing the wireless medium, though it is even difficult to determine the exact number of stations that could access the wireless

media. The most effective way is to reduce the number of collisions caused by data transmission. There are several approaches described in paper [7] are listed below:

- Limit based approach
- Frame based approach
- Local measurement based data control
- Global measurement based data control
- Bandwidth allocation and guard period

The figure below describes the reserved sharing scheme where portion of total bandwidth is allocated to data traffic. With the total bandwidth of T, divides into portions such that each portion carries for a specific traffic flow for e.g.: $\alpha_1 T$ is divided for voice, $\alpha_2 T$ is divided for voice or video, $\alpha_3 T$ if the total bandwidth is reserved for voice for voice, βT is reserved for the guard period which functions as a guard bandwidth reserved partially for collision and idle time to prevent bandwidth allocation from over-provisioning such that $\alpha_1 T + \alpha_2 T + \alpha_3 T + \beta T = 1$.

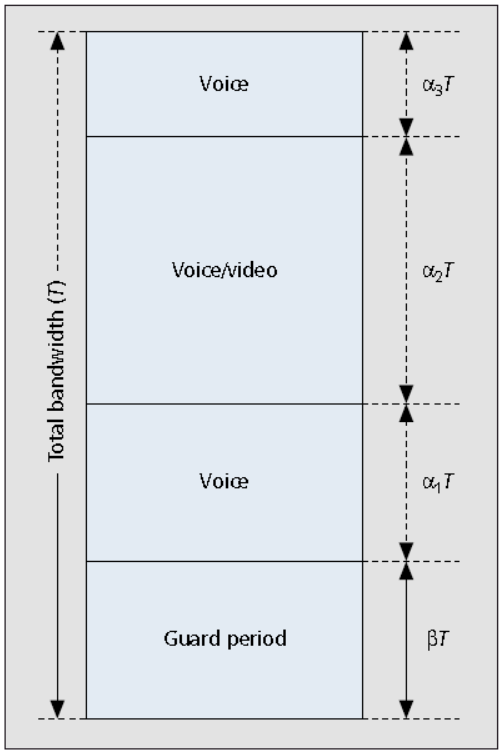

Figure 16: A reserved sharing scheme

Here the goal is to produce designs for QoS guarantee, QoS protection, bandwidth allocation, schemes for content (integrated voice/video/data traffic) delivery in the contention based wireless MAC layer.

Part VII

Conclusion

For the delivery of contents (voice, video and data) in a content delivery network caching, load balancing, compression, quality of service plays a major role. We described various caching techniques that could be used to store contents at various nodes in a network. When caching contents at multiple locations over the network t becomes important which cache server the user is directed to, also to maintain consistency of data across multiple cache servers. Load balancing techniques helped us direct traffic at various webservers such that no webserver is overloaded or idle. Hence, when caching is used it becomes mandatory to have load balancers. We didn't focus much on data compression but compression can play a vital role in reducing the amount of data which needs to be transported over the network resulting in quicker data delivery. Quality of Service determines how efficiently the content could be transferred resulting in minimum loss of data. We looked on some parameters determining QoS for a Video Streaming application also we saw a way to achieve QoS for mobility in quadruple play. We studied on how Akami works and how akami akamizes contents, with the use of ESI (Edge side includes) markup language how caching can be well controlled. We just had an overview of what is a triple play with its difference with quadruple play. Link level packet scheduling and packet retransmission for WLAN was seen and their respective outcomes were illustrated. QoS provisioning, guaranteeing and protection was also studied with its benefits in multimedia content delivery. We have seen not just one but several factors resulting in effective delivery of contents over the network.

Future aspects

A high degree of QoS will certainly increase the efficiency of content flow over the unreliable network. Lot of work has been performed in this area as described in [7], the author proposes lot of models from various papers and concludes lot of work can be performed in the wide area of QoS to increase reliable delivery of data by guaranteeing and protecting QoS. With the need of mobility in triple play, quadruple evolved, more services added to quadruple play would definitely be a market for the ISP's work for the Next Play is still in progress which adds more services to quadruple play.

Acronyms

ACK	Acknowledgement
ATM	Asynchronous Transfer Mode
CDN	Content Delivery Network
CSMA/CA	Carrier sense multiple access with collision avoidance
CTS	Clear to send
CW	Congestion Window
CWinit	Initial Congestion Window size
CWmax	Maximum Congestion Window size
CWmin	Minimum Congestion Window size
DCF	Distributed coordinated function
DCPS	Dual Channel Pipelined Scheduling
DiffServ	Differentiated Services
DIFS	DCF Interframe Space
DNS	Domain Name Server
EDCA	Enhanced distribution coordination function
ESI	Edge Side Includes
FDMA	Frequency division multiple access
HCF	Hybrid channel function
HTML	Hyper text markup language
HTTP	Hyper text transfer protocol
IntServ	Integrated Services
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
MAC	Media Access Control
MPLS	Multi-protocol Label Switching
NAV	Network Allocation Vector

One-Phase BTPS	One-Phase Busy Tone Pipelined Scheduling
PCF	Point Coordination Function
PSNR	Peak signal-to-noise ratio
QoS	Quality of Service
RAM	Random Access Memory
RSVP	Resource Reservation Protocol
RTP	Real Time Protocol
RTS	Request to send
SIFS	Shorter Interframe Space
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
TDMA	time division multiple access
TTL	Time to Live
Two-Phase BTPS	Two-Phase Busy Tone Pipelined Scheduling
UDP	User Datagram Protocol
URI	Uniform Resource Locator
WLAN	Wireless Local Area Network
.JS	Java Script Extension

References

[1] Gilbert Held, *A Practical Guide to Content Delivery Networks, First Edition*, 2006, Chapter 1, 4, 5, 6.

[2] Scott Hull, *Content Delivery Networks – Web Switching for Security, Availability, and Speed, First Edition,* February 2006, Chapter 1, 6, 8, 9, 13.

[3] Mukaddim Pathan and Rajkumar Buyya, "*Content Delivery Networks: Overlay Networks for Scaling and Enhancing in the Web",* Grid Computing and Distributed Systems (GRIDS) Laboratory, Dept. of Computer Science and Software Engineering, The University of Melbourne, Australia

[4] *Load Balancing*, Wikipedia, February 2010, available at http://en.wikipedia.org/wiki/Load_balancing_(computing)

[5] William Stallings, *High-Speed Networks and Internets, Second Edition*, September 2008, Chapter 17, 18

[6] Rafael Pereira, "*QoS Parameters for Video Streaming"*, February 2009, available from http://rafaelspereira.wordpress.com/2009/02/13/qos-parameters-for-video-streaming

[7] Yang Xiao, The University Of Memphis, *"Qos Guarantee and Provisioning at the Contention-Based Wireless MAC layer in the IEEE 802.11e Wireless LANs",* IEEE Wireless Communications, Feburary 2006.

[8] The Promise of the Triple-Play, *Riverstone Networks Whitepaper,* available from http://citeseerx.ist.psu.edu

[9] Xue Yang, Member, IEEE, Nitin H. Vaidya, Senior Member, IEEE, and Priya Ravichandran, "*Split-Channel Pipelined Packet Scheduling for Wireless Networks*", IEEE Transactions on Mobile Computing, Vol. 5, No. 3, March 2006.

[10] "*Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*", IEEE Standard 802.11, June 1999.

[11] Sungwon Han and Ikjun Yeom, *"Retransmission Scheduling for Multimedia Delivery over Wireless Home Networks"*

[12] N Kim and H Yoon, "*Packet fair Queuing Algorithms for Wireless Networks with Link Level Retransmission*", Proceeding IEEE Consumer Communications and Networking Conference (CCNC'04), January 2004.