An Analysis on Performance Evaluation of Intermediate Web Traffic applying Web Optimizer

Dewan Juel Rahman[1], Atikur Rahman[2]
[1]*Rajshahi Science and Technology University, Assistant Professor, juelrstu@gmail.com*
[2]*University of Information Technology and Sciences, Atik8894@gmail.com*

Abstract
Usage of Internet is growing rapidly at this time in all over the world. Among all other services, the World Wide Web is the most popular Internet applications. Internet contains various useful information and therefore, the dependency on Internet is increasing day by day. Providing Internet facilities as a tool to complete mundane activities becomes quite common in schools, colleges, government and non-government organizations etc. However, the process of digitalization has not been achieved as desired due to the lack of infrastructure, awareness and proper use of technologies. Developing countries like Bangladesh are still struggling to provide Internet facilities to all other citizens. The availability of Internet at the user end and high expense of Internet data are the main obstacles right now. There are so many technologies working together on server side to handle the thousands of client's requests at minimum possible time. Those thousand requests come back with thousand responses. All of those messages and contents use Internet as their communication medium and those contents are also known as web traffic. Existing Technologies like web cache and Content Data Network (CDN) work to reduce the web traffic by caching it locally to serve it later. There are few scopes to improve the current technologies. The goal of this thesis is to reduce the web traffic that will lead to faster web browsing experience. Here "Cache Tree" and "LiteWeb" have been introduced and few existing technologies have been modified to work as "Cache Broadcasting" and "Cache Sharing".

***Keywords**:* CDN, Cache Tree, Web Optimizer, Web Traffic, Cache Broadcasting, Lite Web, HTTP.

1. Introduction

The web contents are increasing day by day. Hundreds of Giga Bytes are traveling this internet in each second. To serve better server are sending larger contents like images, videos to clients. Cache is a smart solution to reduce those static contents by saving those in local computer's memory. There are many different technologies such as Content Delivery network, proxy cache server to improve the overall performance of internet. According to HTTP Archive, amongst the top 300,000 sites (by Alexa rank), nearly half of all, the downloaded responses can be cached by the browser, which is a huge savings for repeat page views and visits! Of course, that does not mean that your particular application will have 50% of resources that can be cached: some sites can cache 90%+ of their resources, while others may have many private or time-sensitive data that cannot be cached at all. Fetching something over the network is both slow and expensive: large responses require many round-trips between the client and server, which delays when they are available and can be processed by the browser, and also incurs data costs for the visitor. As a result, the ability to cache and reuse previously fetched resources is a critical aspect of optimizing for performance. Few existing technologies help to improve the web browsing experience. Among them Web Cache and Content Delivery Network are most common technology used by developers. Every browser has HTTP cache management architecture by default. It has to be ensured that each server response provides correct HTTP header directives to instruct the browser on when and for how long the response can be cached by the browser. Together, freshness and validation are the most important ways that a cache works with content. A fresh representation will be available instantly from the cache, while a validated representation will avoid sending the entire representation over again if it has not changed. Figure 6shows a generic view of cache control. When the server returns a response it also emits a collection of HTTP headers, describing its content-type, length, caching directives, validation token, and more. For example, in the above exchange the server returns a 1024-byte response, instructs the client to cache it for up to 120 seconds, and provides a validation token ("x234dff") that can be used after the response has expired to check if the resource has been modified. Web cache helps to reduce the web traffic by serving content locally shown in Figure 9, but the browser has to check the validation each cacheable content separately to be ensured that he is serving the right contents.

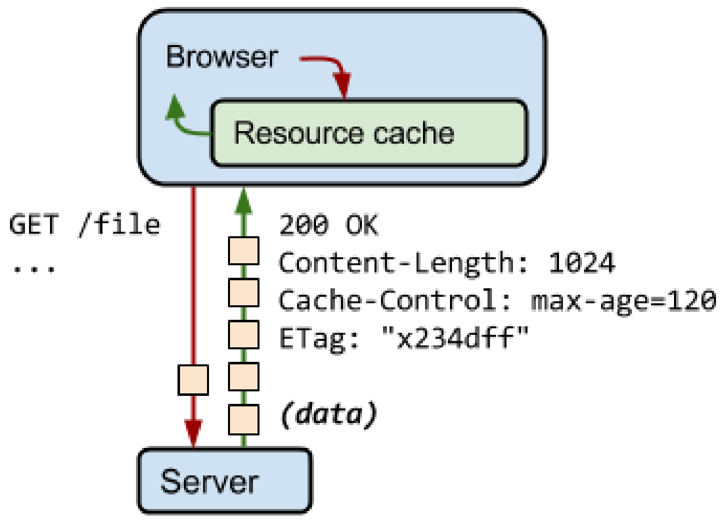


Figure 1: HTTP Response message form server

Web cache can save a huge amount of web traffic, where a cache validation is required as shown in Figure 7. Web cache reduces the amount of downloadable content and it helps to browse website faster. On the other hand, CDN work with the distance between client and server. The goal of "web optimizer" is to optimize the present technology by decreasing the cache validation request through "Cache Tree", "Cache Broadcasting" and let users browse key content based on their bandwidth limitation by using "LiteWeb" and "Cache Sharing" will reduce the web traffic by sharing and resolving cache request locally.

2. Floorplan & Material

Authors World Wide Web is the heart and the internet is the backbone of virtual world. The Web is growing day by day. Simultaneously the web traffic is also increasing with respect to the internet users. There are four different modules of Web Optimizer and their goals are:

1. Cache Sharing: To make reusable-cached content by p2p sharing them with other clients.
2. Cache Tree: To reduce the freshness check request by grouping cache content.
3. Cache Broadcasting: To Reduce network traffic by broadcasting public cacheable content.
4. LiteWeb: Responsive Web Content based on the bandwidth client's PC.

2.1 Proposed architecture:

This Web Optimizer will work with traditional Client-server architecture. It has two parts:

1. Server-Side Web Optimizer (SSWO)
2. Client-Side Web Optimizer (CSWO)

Both will act as proxy server with few additional responsibilities.

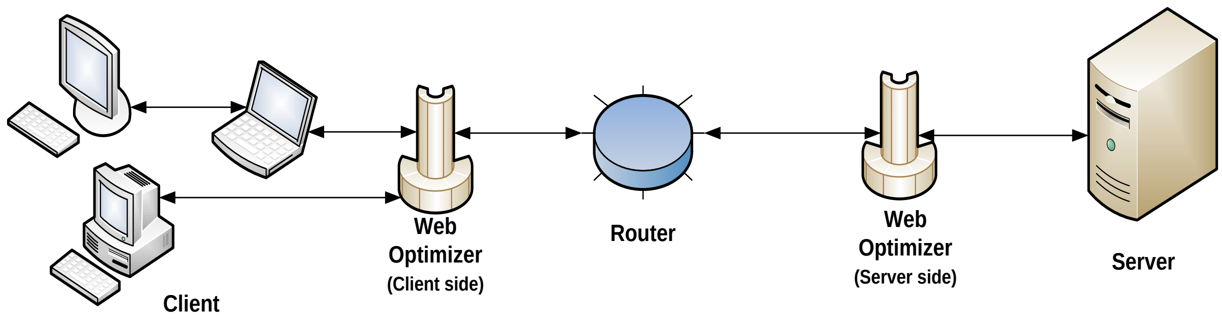


Figure 2: Client side and server-side web optimizer

This Client-side web optimizer will work as proxy server. CSWO will track the public cacheable contents at client side and SSWO will help to broadcast Long Live public Cacheable Contents. Web cache is one of the most used web optimization technique to reduce the web traffic. Storing be contents to use it during next request for that website is known as "Web cache". Content Delivery Network (CDN) is another web optimization technique that stores a copy of all files from server. The architecture of Web cache and CDN has been discussed on next chapter.

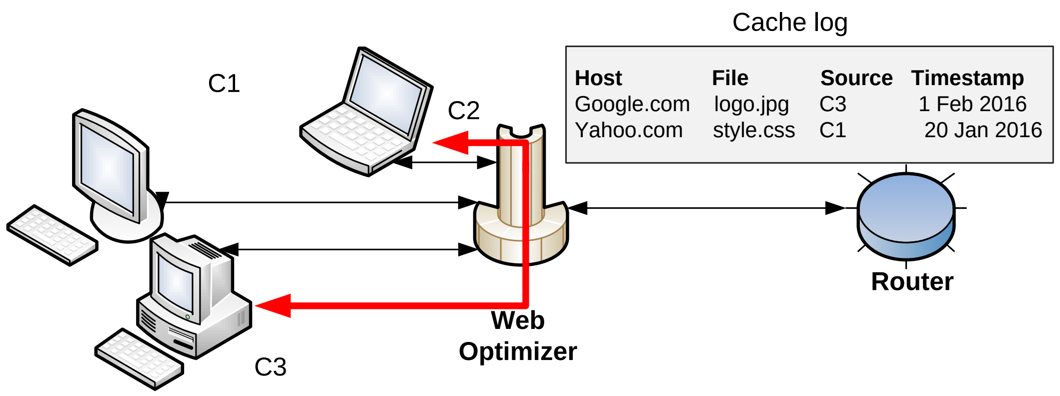
3. Proposed System Design

The goal of Web optimizer is to reduce the intermediate web traffic. There are four different of Web Optimizer and their goals are:
1. Cache Sharing
2. Cache Tree
3. Cache Broadcasting
4. LiteWeb

3.1 Cache Sharing

The aim of this "Cache Sharing" is to decrease the distance between server and client by serving the content from a neighbour's computer. It will improve the browsing experience by decreasing the downloading time and it will decrease the web traffic. Client-side web optimizer (CSWO) will keep track of cacheable content of client's PC. If CSWO receive a new request that could be served from client side, then it will not send the request to the server. CSWO will connect host pc (who has that content) with the requester and establish a peer-to-peer connection to transmit the requested content. The goal "Web Optimizer" could be achieved by with the help of Client-side web optimizer. Shareable browser cache will reduce the load of proxy server, network traffic and web pages will be faster. Normally Browser caches are stored in user's computer. The aim of "Cache Sharing" is to make shareable those caches with others.

The CSWO have to maintain a log of public cache contents of end user. A Browser plug-in "Cache Manager" could be introduced to handle intra-networked Figure 5 cache requests shows how the proposed architecture will work. Here the CSWO is playing a vital role in this architecture. CSWO will check the availability of cache provider, which have the requested cache and alive on the local network. If there are any, then it will set a peer-to-peer connection with client and cache provider. The CSWO is responsible to check the freshness of cache by using the E-Tag or/and modification date of that contents. If the cache failed to pass the validation test, then the cache will be fetched from main server. This cache sharing technique will reduce the web traffic. Less web traffic and request is needed for faster internet.



Host	File	Source	Timestamp
Google.com	logo.jpg	C3	1 Feb 2016
Yahoo.com	style.css	C1	20 Jan 2016

Figure 3: Cache log of CSWO

3.2 Cache Tree

The aim of "Cache Tree" is to find out an efficient and effective solution to reduce web traffic by optimizing cache validation. "Regular Cache Tree" will contain all cacheable contents of a website. This tree will be generated based on the file types. The data structure of cache tree is very important. According to the aim of "Cache Tree" the most updated contents have to be on the top of the tree. Therefore, that most recent file will get higher priority than others will. The Cache tree is heap where recent content will get higher priority.

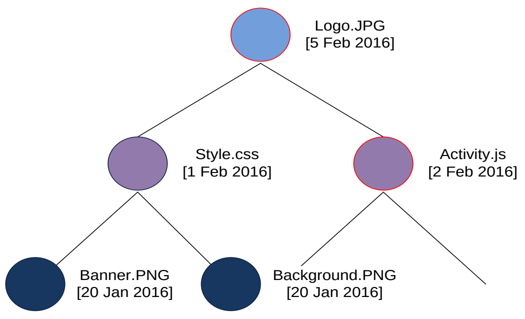


Figure 4: Regular Cache Tree (Server Side)

Same cache tree will be maintained in server side and client side. And the cache tree of client side will be synchronized by downloading the new content from server.

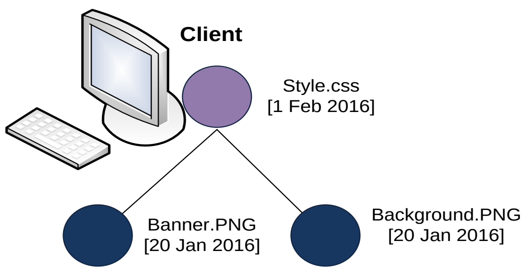


Figure 5: Cache Tree (Client side)

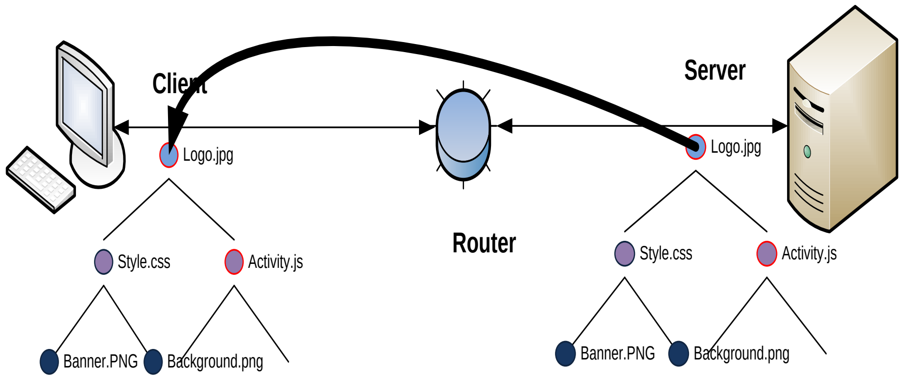


Figure 6: New contents have been served from server to client

3.3 Cache Broadcasting

The aim of "Cache Broadcasting" is to reduce the intermediate web traffic by broadcasting Public cacheable contents. The goal of Cache broadcasting is to reduce the intermediate traffic of server and client with the help of SSWO and CSWO. Here LLpCC (Long Live Public Cacheable Content) are suitable contents for broadcasting, because they do not change very frequently and could be cached for a long time. So the broadcasting requests have to be called very frequently. Server-Side Web Optimizer will handle this cache broadcasting with the help of client-side web optimizer. Server- side web optimizer (SSWO) is responsible for broadcasting LLpCCs. Server will send the broadcasted content to the SSWO and it will send all contents to all known Client-Side Web Optimizer (CSWO). CSWO have to keep track of all end users who have browsed that content earlier and contain those LLpCCs. CSWO receive the broadcasting request from SSWO and then send it to end user. CSWO will keep those LLpCC and during the next request for this content then CSWO will serve it. During regular browsing, the CSWO is responsible to check the freshness of those LLpCC. Cache Tree could be implemented to check the freshness by only one request to be ensured that all LLpCC are fresh or CSWO will check the freshness periodically (once a day) According to the old system they have to send individual request for each content. In most of the cases, those responses are negative. This broadcasting will reduce the server to client data transmission.

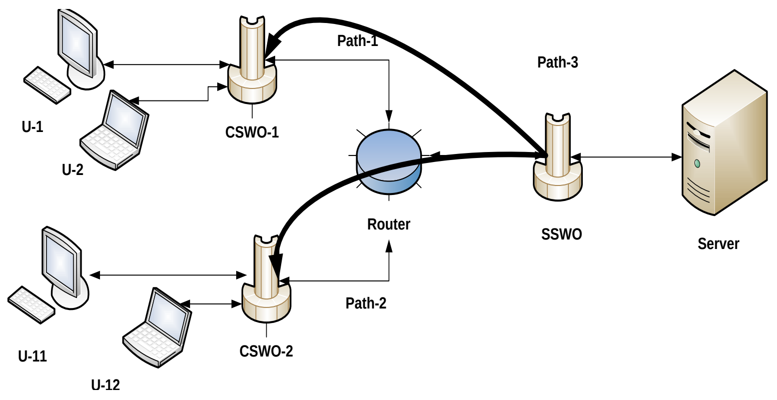


Figure 7: SSWO Broadcasting Cache to CSWO

3.4 LiteWeb

LiteWeb will be helpful to browse a website with low bandwidth. Many web sites contain so many style sheet, animations, and videos to make web pages more attractive to the viewers. Except key-contents most of them are avoidable. A web page that contains only key-content will be treated as example of LiteWeb. Client-side Web optimizer will send request for LiteWeb content based on the bandwidth of clients.

Clients may also change the settings as

- Automatically detect and request for LiteWeb.
- Only LiteWeb
- LiteWeb (key content) will get higher priority

CSWO will also keep track of client's bandwidth and send it to the SSWO. SSWO will check the bandwidth of client-side computer. It will check the settings (embedded information with request) from the request of the web browser, then SSWO will take decision based on the client's preference.
During regular transmission key content will get higher priority and will be sent before other contents

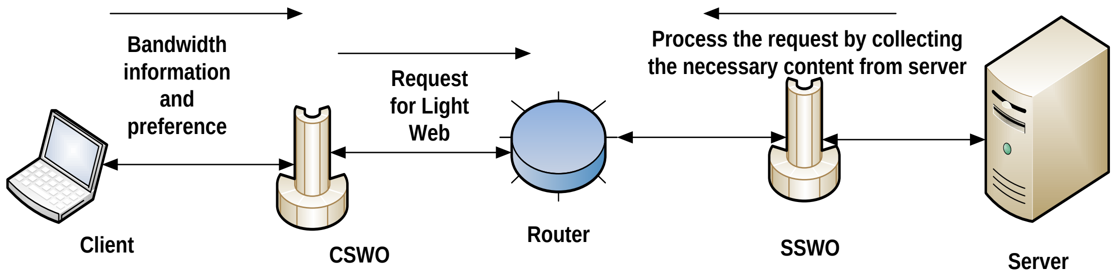

Bandwidth information and preference
Request for Light Web
Process the request by collecting the necessary content from server
Client
CSWO
Router
SSWO
Server

Figure 8: server is serving "LiteWeb" request

This "LiteWeb" will reduce the web traffic that will help the client to download necessary content faster within limited bandwidth. Key contents will get more priority, so the clients do not have to download unnecessary scripts or images. The LiteWeb will also reduce the web traffic.

4. Validation

To validate the proposed solution a prototype system has been designed based on java. Where the server sends response to the client according to the proposed architecture.

4.1. Prototype of Cache Tree

A prototype has been designed to demonstrate the working procedure of "Cache Tree" based on java (Appendix I). Where, there are 12 contents on the server. Client has required those contents to render a page. During the first request, the client will download all of those contents

```
From Client: Have to update items(id): 0 1 2 3 4 5 6 7 8 9 10
11
From Client: Adding Content: 0
From Client: Adding Content: 1
From Client: Adding Content: 2
From Client: Adding Content: 3
From Client: Adding Content: 4
From Client: Adding Content: 5
From Client: Adding Content: 6
From Client: Adding Content: 7
From Client: Adding Content: 8
From Client: Adding Content: 9
From Client: Adding Content: 10
From Client: Adding Content: 11
```

Figure 9: Client is Downloading content from the server (First time)

There is a heap on the server side and it works as “Cache Tree”. This heap contains the content id and timestamp.

```
Heap Content: Id: 9 Time: 1
Heap Content: Id: 10 Time: 1
Heap Content: Id: 11 Time: 1
Heap Content: Id: 6 Time: 4
Heap Content: Id: 8 Time: 2
Heap Content: Id: 5 Time: 5
Heap Content: Id: 4 Time: 6
Heap Content: Id: 0 Time: 10
Heap Content: Id: 3 Time: 7
Heap Content: Id: 2 Time: 8
Heap Content: Id: 7 Time: 3
Heap Content: Id: 1 Time: 9
```

Figure 10: Heap data structure has been used to construct this Cache Tree

If there are no updates on the server side, then it will send a negative response to the client.

```
From Client: Nothing to do, updated Till now
```

Figure 11: There is no updated content on the server

Meanwhile, two new contents have been added on the server and two other contents have been updated. At the same time, the cache tree has to update accordingly. The cache tree will help the server to find out the new content at minimum possible time and inform the client.

```
Heap Content: Id: 12 Time: -1
Heap Content: Id: 7 Time: -1
Heap Content: Id: 13 Time: -1
Heap Content: Id: 6 Time: 4
Heap Content: Id: 10 Time: 1
Heap Content: Id: 5 Time: -1
Heap Content: Id: 9 Time: 1
Heap Content: Id: 0 Time: 10
Heap Content: Id: 3 Time: 7
Heap Content: Id: 2 Time: 8
Heap Content: Id: 8 Time: 2
Heap Content: Id: 1 Time: 9
Heap Content: Id: 11 Time: 1
Heap Content: Id: 4 Time: 6
```

Figure 12: Updated “Cache Tree” of Server

After that, if client request for those content. The server will inform the client that there are 4 new updates.

```
From Client: Have to update items (id):  5 7 12 13
```

Figure 13: Server informed the client about new contents

After that, the client will collect those contents form server.

```
From Client: Updating Content ID: 5
From Client: Updating Content ID: 7
From Client: Adding Content: 12
From Client: Adding Content: 13
```

Figure 14: Client collect the new updated content form server

If the modification date is same or older than no data will be send. A negative response will be send. (12 different request and response messages have been resolved by only one request and response message. Here it also saves the server processing time) File compression technology could be used to reduce more web traffic. If there are more than one newer file than we may send only one compressed file

4.2. Prototype of Cache Tree

A prototype has been designed to demonstrate the working procedure of “Key Content”. To identify the key content an extra information has been added with content properties. If the client request for “Key contents” due to bandwidth limitation, then the server will send only the key contents related with the requested page.

```
Number of contents :5
Total Data: 3800
File name: logo.jpg || Size: 250 || Modification Date: 5
File name: style.css || Size: 50 || Modification Date: 5
File name: image1.jpg || Size: 500 || Modification Date: 5
File name: image2.jpg || Size: 1000 || Modification Date: 5
File name: image3.jpg || Size: 2000 || Modification Date: 5
```

Figure 15: Regular Request served by Server

Here the server sends five contents and the total size of five contents are 3800 Bytes. If the client request for only “key contents” then the server will send only two contents. Here, this procedure saved 3050 bytes and it will load five times faster than regular system

```
number of contents :2
Total Data: 750
File name: logo.jpg || Size: 250 || Modification Date: 5
File name: image1.jpg || Size: 500 || Modification Date: 5
```

Figure 16: Key Contents Severed by the server

Conclusion

The goal of web optimizer is to minimize the intermediate web traffic that will help to browse web content faster. Few excising architectures work to improve the web browsing experience. Among them Web Cache works to minimize the number of downloadable contents and Content Delivery Network (CDN) works to minimize the distance between client and server. The working procedures of web cache and CDN have been discussed on chapter 2. Where web cache requires freshness check for each cached content and it causes extra web traffic. Redundant web content on internet is another scope of improvement. A scenario has been discussed on chapter 1, where redundant content exit on intermediate path in between server and local network. Bandwidth limitation on client’s end is one of the major challenges of present web technology. A new architecture named as “Web Optimizer” has been proposed to improve the present scenario of web. This architecture includes Server-Side Web Optimizer (SSWO) and Client-Side Web Optimizer (CSWO) as transparent proxy server. This SSWO and CSWO helps to minimize web traffic by using “Cache Tree”, "Cache Broadcasting" and "Cache Sharing". To serve the key contents to the clients "LiteWeb" have been introduced. Cache Tree helps to synchronize cached content of browser with server's contents based on the modification date. Only new or updated contents will be downloaded from server. Only one check is required to be confirmed that there are no downloadable contents on the server, where the number of cached contents does not matter. Cache Broadcasting ensures that multiple clients of a network do not have sent same request to the server for required content individually. The server will broadcast new cacheable contents to all known CSWO. Then CSWO is responsible to distribute or serve it to the client. Cache Sharing tries to collect required contents from local network. Here CSWO maintain a log of public cacheable contents. If any client of that network requires any file, then CSWO check its log and if that content is stored on another computer of same network then the CSWO establish a peer-to-peer connection with another. Then the host of that content acts as server and send the response. Here CSWO is responsible to check the freshness of those contents according to its log file. LiteWeb helps to serve only the key content based on the request of client. This LiteWeb helps to browse website with low bandwidth. The proposed "Web optimizer" will help to browse internet faster by minimizing the web traffic.

References

[1] Mukesh Dawar, Charanjit Singh , "Study on web Caching Architecture: A Survey", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 11, November 2013.
[2] Valeria Cardellini , Michele Cola janni, Philip S. Yu , "Dynamic Load Balancing on Web-server Systems", IEEE Internet Computing, vol. 3, no. 3, pp. 28-39.
[3] "Optimizing content efficiency", URL: https://developers.google.com. Accessed on: 20January 2016
[4] "How to control cache", http://www.web-caching.com/, Accessed on: 22January 2016
[5] "Caching tutorial", https://www.mnot.net, Accessed on: 29 February 2016.
[6] "Caching in HTTP", https://www.w3.org/Protocols/, Accessed on 2 February 2016.

Dewan Juel Rahman, born in Narayangonj, Bangladesh. He obtained his Bachelor's in Electronic and Communication Engineering (2011) from University of Information and Technology, Bangladesh and M.Sc. in Computer Science (2014) from Jahangirnagar University, Bangladesh. Dewan Juel Rahman is an Assistant Professor and Head of the Department of Computer Science and Engineering at Rajshahi Science and Technology University (RSTU), Natore, Bangladesh from 2016. From 20013 until 2016 he was a Lecturer of the Computer Science and Engineering Department at the same University. Dewan Juel Rahman was working as an Information and Communication Technology specialist at University of Information Technology and Sciences (UITS), Dhaka, Bangladesh. His field of interest is Software Engineering, Information Technology and Networking.

Atikur Rahman, born in Narayangonj, Bangladesh. He obtained his Bachelor's in Electronic and Communication Engineering (2011) from University of Information and Technology, Bangladesh. Atikur Rahman was working as an Information and Communication Technology specialist at University of Information Technology and Sciences (UITS), Dhaka, Bangladesh. His field of interest is Mobile Communication, Information Technology and Computer Networking.