International Journal of Hybrid Information Technology
Vol. 9, No.8 (2016), pp. 43-52
http://dx.doi.org/10.14257/ijhit.2016.9.8.05

Implementation of Server Handover for Live Streaming Application during High Traffic with Bitrate Adaptation

Rajina R. Mohamed[1], Wahidah Hashim[1], Ahmad Fadzil Ismail[2], Khalid M. Abdilahi[1] and Mohammad Kamrul Hasan[2]

[1]Department of System and Networking, Universiti Tenaga Nasional, Kajang, Selangor, Malaysia
[2]Department of Electrical and Computer Engineering, International Islamic University Malaysia, Gombak, Selangor, Malaysia
Rajina@uniten.edu.my; Wahidah@uniten.edu.my; af_ismail@iium.edu.my; hasankamrul@ieee.org; khalidcawl09@gmail.com, hasankamrul@ieee.org

Abstract

As the rate of internet users is increasing on a daily basis, a tremendous amount of these users are accessing online streaming websites and applications. This paper proposes a live streaming architecture that supports traffic balancing during peak times and bitrate adaptation. Rather than replicating content on multiple servers, we implemented traffic balancing by assigning a temporary regional server to manage the traffic at regional area. Bitrate adaptation at server side was also proposed and implemented to provide a fair video quality to all users. Through a few tests that has been run on the testbed, we can see the live video streaming was move seamlessly from one server to another server with an acceptable video quality among the users.

Keywords*: traffic balancing, Live streaming, bitrate adaptation, Server handover*

1. Introduction

Video streaming is one of the most common uses of the internet today [1]. Users who have different devices such as laptops, mobile phones, and tablets access the internet to use streaming applications. These applications can be web-based streaming such as Wowza media streaming [2] or desktop based such Plex [3]. As long as the technology is advancing and internet speed is increasing, these users are demanding high quality real-time video and multimedia transmissions. This imposes a challenge on streaming content providers. One of the challenges is traffic management when the number of users who are simultaneously accessing their streaming servers increase. During peak times in a specific region or area, the number of users increase exponentially to the extent that streaming servers get overloaded and reduce the server's bandwidth. The quality of the streamed video also will affected due to the loss or delay of frames caused by server overload. In this paper, we propose an architecture that employs effective load balancing technique to handle the load during peak times. The architecture also makes use of bitrate adaptation mechanism provided by HTTP live streaming protocols and use the protocol in a unique way. The novelty of this architecture is the temporary usage of the local streaming server. In the previous work that we have discussed in the literature review section, most architectures implement load balancing by making the local server a duplicate of the primary server which contains the same content. In contrast, our local streaming server re-streams from the primary server and does not have any video content to steam

ISSN: 1738-9968 IJHIT

2. Problems during High Traffic and the Solutions

In this section, we will discuss the problems with the current streaming architecture and the proposed architecture.

Figure 1 below shows the current streaming method used by service providers. It encompasses architectures that use P2P, CDN, and Replica streaming servers which are impractical for some organizations, costly, and need expensive maintenance [4-6]. This is just the simplified version of these architectures. The emphasis here is to show that the streaming is always done directly from a public streaming server on the internet. Other details and components such as edge servers in CDN networks, Replica servers, and so on are hidden from this simplified diagram.

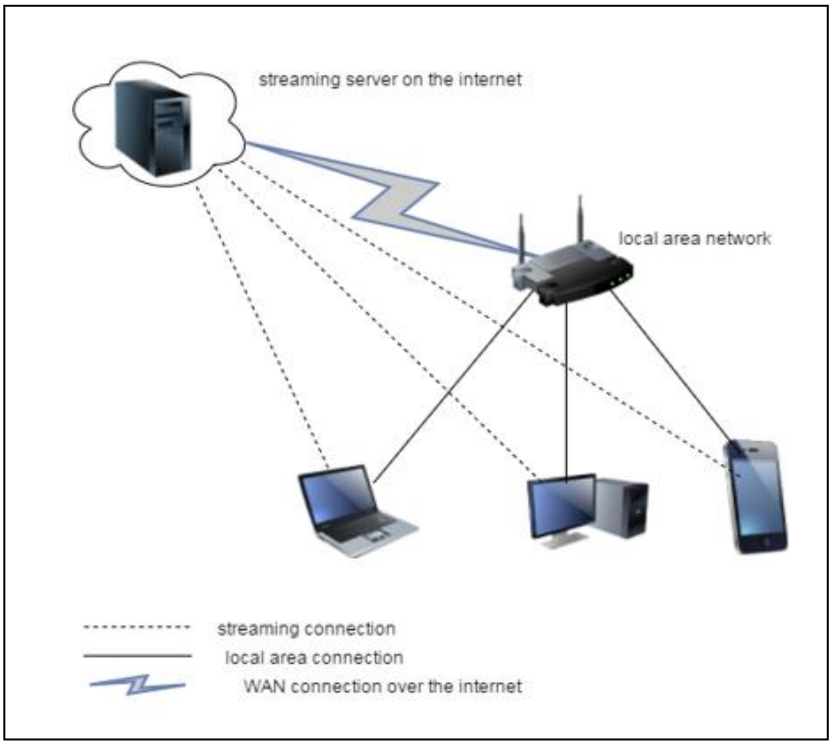

Figure 1. Current Streaming Approach

As show in the figure above, all of the clients in the local area network are directly streaming from the public streaming server. If the number of clients increase in this area, they would still be accessing that server and overload it. In some architectures, some of the clients are redirected to a replica server [7]. However, such resources get exhausted quickly during peak periods.

3. Proposed Architecture

In our proposed architecture, there are three main components: Central Server, Traffic Coordinator, and Regional server. The Central server consists of the video content in a segmented format which we will discuss later. Anyone that accesses the Central Server will be given those segments (video on demand or live). The Regional Server basically will serve client base on region. It does not contain any video data. The Regional Server streams from the Central Server (as a client) and re-streams to the clients in its region (as a server). Therefore, it is a matter of getting the stream from the primary server and forwarding it. The Traffic Coordinator is the server will monitor the traffic at the Public Server and will measure the either the network traffic exceed the threshold or not.

Clients and Regional servers communicate with the Traffic Coordinator before they stream anything. The Traffic Coordinators registers them (clients and Regional Servers) and tells them where to stream from. In the case of Regional servers, they will also stream from the Central Server. In the case of clients, the Regional Server will stream from the Central Server if they haven't exceeded their threshold. Otherwise, they will be redirected to their assigned Regional Server. Figure 2 below shows how the architecture works during normal traffic periods. Figure 3 shows what happens when high traffic comes from one of the regions.

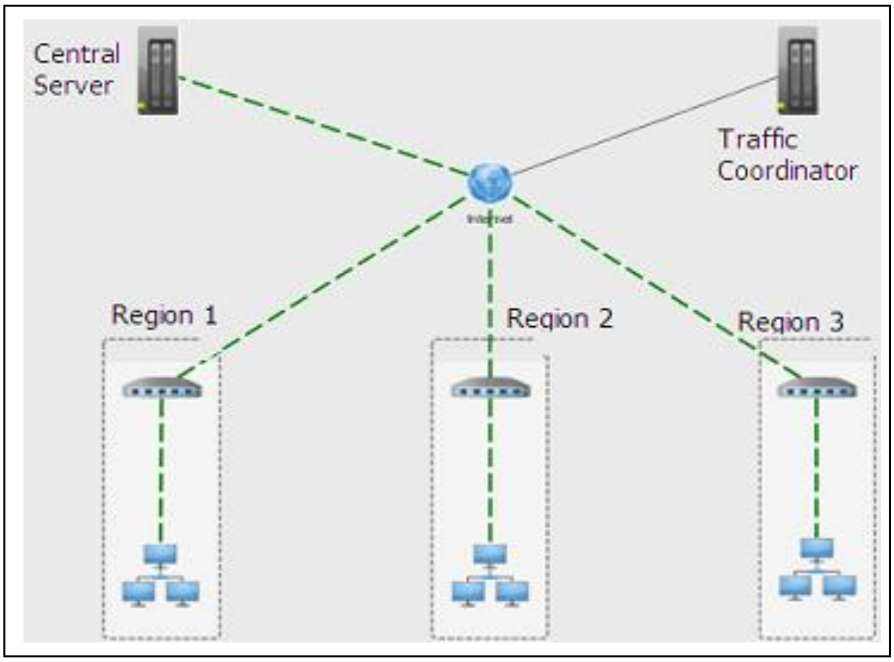

Figure 2. Proposed Architecture – Normal Traffic Scenario

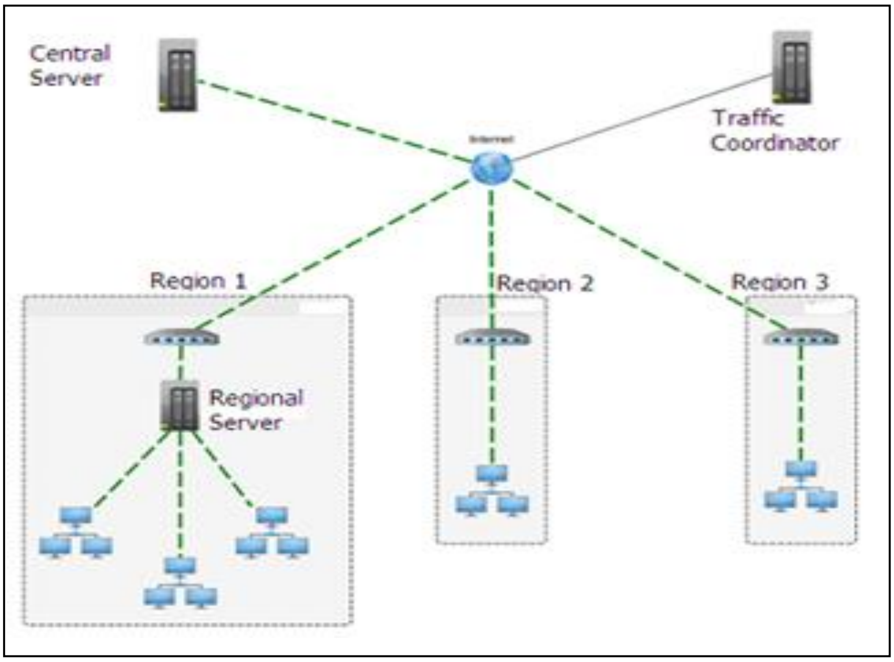

Figure 3. Proposed Architecture – High Traffic Scenario

2.1. Traffic Flow Diversion

Traffic flow diversion has been done in two ways in this study. First, the Central Server is always kept from getting overloaded by the Traffic Coordinator by always checking the threshold value. Since clients come to the Traffic manager first, it keeps a list of those connected to the Central Server and another list of those connected to the Regional Server. The Central Server will not get overloaded as long as the threshold value is not reached.

Second, the Traffic Coordinator keeps an eye those clients who get disconnected from the Central Server. For example, if the threshold for region A is 50 and all 50 positions are occupied, any client that comes from region A will be redirected to their Regional Server. If a client from region A disconnects from the Central Server (49 from A), it will be replaced by a client from A that was initially connected to the Regional Server. This always makes sure the Central Server is fully utilized while balancing its traffic and also the Regional Server is fully utilized while balancing its traffic as well. Therefore, the traffic balancing happens fairly between the Central Server and the Regional Server.

It is fundamental to understand that this local server is just a temporary server. This means that it serves as a temporary stream relay for a short period of time (during high traffic period) and serves its users only during this interval of time. This server is not a replica of the primary server which means it does not have a duplicate content of the primary server. Instead, it streams from the primary server and then re-streams to its clients in real-time. This is one of the most fundamental differences between this architecture and the exiting architectures.

2.2. Bit Rate Adaptation

There are two integrated methods that we have used to implement this architecture. First, the streaming protocol that we used is Apple's HTTP Live Streaming (HLS) [8]. Second, we have written Java programs that make use of the VLCJ API [9]I to implement the streaming between servers and clients.

HLS is one of the widely-adopted streaming protocols that use the HTTP protocol and support bitrate adaptation. Bitrate adaptation allows the streaming clients to choose what quality of video to stream from the server based on their connection strength [10] . If the video requesting client's connection is not good, it will request low quality video from the server, and vice versa. This is very important for those clients that are connected to wireless connections where the signal strength fluctuates frequently

In conventional method, HLS achieves this through two main components: the stream segmenter, and the distribution server. When a live or on demand video is produced, the stream segmenter segments the video into fragments or segments. Each segment is encoded with the MPEG2-TS protocol to be transported over the network. The segments also generates different bitrate (quality) fragments (*e.g.* 128Kbps, 1024Kbps). These segments of different quality are stored in a standard HTTP server such as Apache.

Before a client requests a video segment, it will detect its available bandwidth and request the video quality that suits this bandwidth. However, our usage of this protocol is different from the conventional or normal approaches. Rather than allowing the clients to decide what video quality to get from the server, the Central Server decides what video quality to stream. This means that we are doing bitrate adaption from the server side rather than the client side. In situations where all or most of the clients have very good connections, they will request the high quality content from the server. Consequently, the server will be left with small bandwidth. This creates network congestions and insufficient bandwidth on the server side. Therefore, the clients will experience packet loss, delay, and jitter. To avoid this, the streaming server must adjust the quality of video it is streaming based its available bandwidth.

As far as the clients are concerned, it will not have much effect since it is still streaming through HTTP and HLS. It has the same effect as serving a single video bitrate in the normal approaches. The client can still download video segments while the user is watching the previously watched ones.

3. Testbed Implementation

Our testbed implementation comprises an implementation at both servers and clients. The server components implemented are the Central Server, Regional Server, and the Traffic Coordinator. As mentioned earlier, VLCJ API was customized in order to communicate with the native LIBVLC API of VLCJ media player. As for the clients, VLCJ API was also used but only to fetch the stream and display it for the user.

3.1. Server and Client Setup

All the hardware for servers were installed and configured by Linux-based operating system. The Central Server was equipped with Apache Server as well as video contents as to be accessed by clients. Traffic Coordinator is equipped with specific algorithm to monitor all the connectivity between client and servers. It will keep all the information about clients, its' IP addresses, URL addresses that requested by clients and clients' domain. Traffic Coordinator will monitor the traffic condition between clients and Central Server by keeps track of the number of clients from each region and continuously checks this number against the threshold. If the number of clients exceed the threshold, any new client from that area will be redirected to their local server.

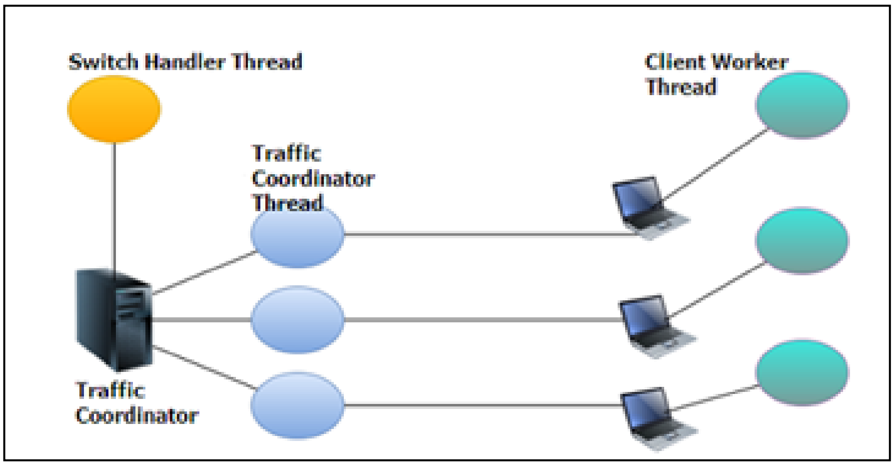

Figure 4. Thread Components of Traffic Coordinator and Clients

It retrieves the primary server's streaming URL (*e.g.* http://192.168.1.102:8080/). It then establishes connection with the local server and gets its streaming URL as well. Once the clients come to get connected, the traffic manager registers them and gives them the stream URL (primary or secondary depending on the threshold). The client then used the API to fetch the video segments from the assigned server and display it for the user.

To make communication efficient, the traffic manager assigns a thread for each client connected to it. This thread gives the stream URL to the client and waits for its shutdown. In addition to that, the traffic manager has a switch hander thread.

This thread takes a client away from the local server and assigns it to the primary server to achieve load balancing. Figure 4 shows the component of the Traffic Coordinator.

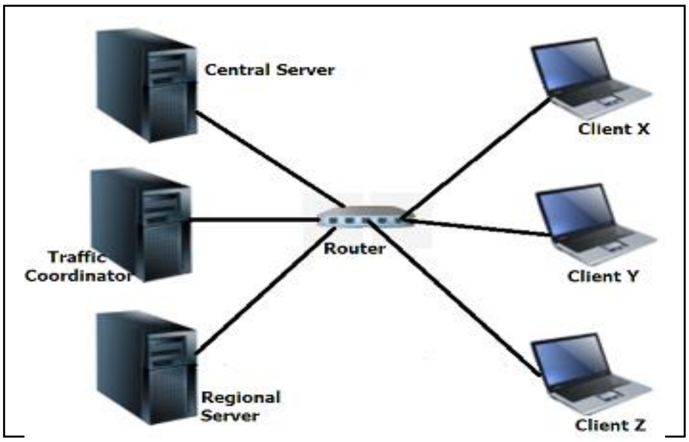

Figure 5. Layout of our Testbed Setup

As for the Regional Server it was equipped and configured with the same environment with the Central Server but need only to serve regional’s client. Regional Server is a temporary server, only activates during high traffic and deactivates when traffic condition is low. While for clients, were installed with Linux-based environment with VLCJ client in order to get video streaming from the server. Generally, the testbed layout as portrayed. The real testbed as shown in Figure 6, which consists only three laptops responsible as Central Server, Traffic Coordinator and Regional Server respectively. To simplify things, all the three laptops were also used as clients throughout the time

5. Testing Result and Discussion

The threshold must be specified on the Traffic Coordinator before running it. For example, a threshold of 3 means anything more than 3 clients will be redirected to their Central Server. As described earlier, clients get their stream URL from the Traffic Coordinator based on the threshold. Clients also get redirected from the Regional Server to the Central Server if any client disconnects from the Central Server. There are two major objectives that this architecture is trying to achieve; traffic balancing and bitrate adaptation. To get the consistence result, the system has been run and tested more than 20 times to get the consistence result.

Figure 6. Actual Hardware and Layout for the Testbed

Figure 7 above shows two clients streaming from two different servers when the threshold was set to 1. First client is streaming from the Central Server (192.168.1.102) while the second client is streaming from the Regional Server (192.168.1.105). The video content is streamed synchronizing between both Central Server and Regional Server.

Figure 7. Traffic Balancing Result

In order to achieve the second objective, we have to see the effect of video quality streamed by clients when the traffic is high. To realize this, we test by using Apache's ab tool [11] to simulate traffic on the web server. When a lot of traffic was generated, the server selected its next lowest bitrate quality and streamed to the client. Figure 8 below shows the primary server selecting the next lowest bitrate (28kb) to adjust the video quality.

Figure 8. Bitrate Adjustment for Video Quality

When the server was not busy and streaming high quality video, the clients were able to stream it without any issues except when their connection was slow which resulted in startup delay of few seconds as expected.

9. Conclusion

We proposed an architecture to solve the high traffic problem during peak times by assigned a temporary regional server for streaming service application. High traffic condition might happened during an important or popular events *i.e.* sports events, presidents' speeches, and so on. In these scenarios, the deployment of an architecture that enhances traffic balancing and efficient use of resources is very crucial. Our proposed architecture addresses this issue. We also deployed HTTP Live Streaming (HLS) protocol for bitrate adaptation implementation to provide a better and fair quality among clients. Testing has been done to see the capability of Regional Server take responsibility to cater their own regional clients. We also test the bitrate adaptation method provided by the Central Server in order to offer a fair quality for all clients in the same domain. Our future work is to test the architecture with large-scale clients and to improve the flexibility of bitrate adaptation during high traffic.

References

[1] Internet Live Stats, "Internet Users", [Online]. Available: http://www.internetlivestats.com/internet-users/, (2015).
[2] WowZa Streaming Engine, availabe at https://www.wowza.com/products/streaming-engine.
[3] Plex, accessible at http://www.plex.com.
[4] S. Xu and H. Shen, "QoS-oriented content delivery in e-learning systems", IT in Medicine & Education, 2009. ITIME '09. IEEE International Symposium on, vol. 1, **(2009)**, pp. 665–670.
[5] S. I. Lee, D. Lee and S. G. Kang, "Challenges to P2P live video streaming over mobile heterogeneous networks", Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on, **(2011)**, pp. 36–41.

[6] M. Gupta and A. Garg, "Content Delivery Network Approach to Improve Web Performance: A Review", International Journal of Advance Research in Computer Science and Management Studies, vol. 2, Issue 12, **(2014)**.
[7] J. Kim and Y. Won, "Dynamic load balancing for efficient video streaming service", Information Networking (ICOIN), 2015 International Conference on, **(2015)**, pp. 216–221.
[8] A. Fecheyr-Lippens, "A Review of HTTP Live Streaming Table of Contents", Memory, no. January 2010, **(2010)**.
[9] T. C. Thang, H. T. Le, A. T. Pham and Y. M. Ro, "An evaluation of bitrate adaptation methods for HTTP live streaming", Sel. Areas Commun. IEEE J., vol. 32, no. 4, **(2014)**, pp. 693–705.
[10] L. Lambrinos and E. Demetriou, "An adaptive live media streaming architecture", 2nd Int. Conf. Adv. Multimedia, MMEDIA 2010, pp. 74–77, **(2010)**.
[11] Apache Bench, accessible at http://infoheap.com/ab-apache-bench-load-testing/.

Authors

Rajina R. MMohamed, She received her bachelor degree in computer science from University Putra Malaysia (UPM). She then pursued her MSc in computer science majoring in distributed computing at the same university and completed her MSc in 2003. She currently works as a lecturer at Universiti Tenaga Nasional (UNITEN), Malaysia since September 2013. Previously, she was a researcher at MIMOS Berhad focusing on computer networking. Since 1996, has involved in various field of research including network information security, wireless communication (IPv4 and IPv6), digital home and e-learning. Rajina has published several publications and filed several patents on her research findings. She is a member of IEEE.

Wahidah Hashim, She received her bachelor degree in Information Technology, Business Management and Language from University of York, UK in 1999. She then pursued her MSc in Multimedia Technology at University of Bath, UK in 2001. She completed her PhD studies from King's College London, UK in 2008 in the field of Telecommunication Engineering. She is currently at Universiti Tenaga Nasional, system and networking department, College of Computer Science and Information Technology as a principle lecturer since 2014. Prior to this, she was a staff researcher at MIMOS Berhad, a Malaysian National R&D in ICT sector. Apart from her main task in doing research in cognitive radio, WLAN, OFDM, MIMO systems, IoT, Big Data and wireless system, she is actively involved in the development of Malaysia technical specification, standard and guidelines of wireless devices, International Mobile Telecommunication (IMT) systems and sensor network for Malaysian Communications and Multimedia Commission (MCMC). She had represented Malaysia at International Telecommunication Union (ITU) working party 5D meeting for IMT-Advanced technology as well as a member to ITU Focus Group on Aviation Cloud Computing and Asia Pacific Telecommunity Standardization Program (ASTAP). Dr. Wahidah has published several publications and filed several patents on her research findings. She is a member of the IEEE and IACSIT.

Ahmad F. Ismail, He is currently serving as associate professor at the Department of Electrical and Computer Engineering, Faculty of Engineering, International Islamic University Malaysia (IIUM). He completed his bachelor degree studies in Electrical Engineering at Gannon University, Pennsylvania, USA with Cum Laude Latin honors. He holds MSc in telecommunications and information systems from University of Essex, UK and PhD in Electronics from University of Bath, UK. His research interests include millimeter and microwave propagation studies, development of active and passive target tracking algorithms and Cognitive Radio (CR) applications. He is a registered Professional Engineer with Board of Engineering Malaysia and also a senior member of the IEEE. He is currently the deputy dean of research management centre of IIUM.

Mohammad Kamrul Hasan, He is currently a PhD candidate in Communication Engineering at the department of Electrical and Computer Engineering in International Islamic University, Malaysia. He achieved his received his Masters in Communication Engineering from International Islamic University, Malaysia in 2012. His current research interests include OFDMA, Interference, Cognitive Network, Optimization, Big Data, Smart Grid Computing, and Data Communication and Networking. He published more than 50 papers in international journals and conferences. He is a Member of Institute of Electrical and Electronics Engineers (MIEEE), and Member of Institution of Engineering and Technology (MIET).

Khalid Muhumed Abdilahi, He is a Graduate Research Assistant at the Universiti Tenaga Nasional (UNITEN) in Selangor, Malaysia. He received his Bachelor's degree from UNITEN in Computer Science major in System and Networking. He has involved research-based final year project to develop an algorithm of traffic diverting by using temporary server. He has developed interest in mobile communication, wireless communication, Grid and cloud computing. He can be contacted directly via email at khalidcawl09@gmail.com.