Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 3159-0040
Vol. 2 Issue 4, April - 2015

Video Streaming Over Full Duplex Network Using Websocket and Its Performance Evaluation

Nikita Kulkarni
Department of Computer science,
University of Bridgeport
nikitagkulkarni@gmail.com

Tarik Eltaieb
Department of Computer science,
University of Bridgeport,
teltaeib@my.bridgeport.edu

***Abstract*—This paper, describes the use of websocket for video streaming, which enables a two-way communication between clients in a flexible and short-time manner. In websocket protocol, first an opening handshake is made and then message framing is done over the TCP connection. This technology provides mechanism of two way communication between servers and browsers that do not want multiple opening of HTTP connections. This functionality supports browser-based application. Security mode used for websocket protocol is the original model which is used by most if the web browsers. The advantage of websocket over HTTP is that, a long lasting connection can be established over a single request. This is done on a persistent TCP/IP socket. Websocket was originally introduced for real-time applications such as gaming, chats etc [1]. In this paper we can see the use of websocket for video streaming.**

In today's fast growing world of technology, one needs faster communication. And websocket provides faster and effective video streaming over the traditional TCP protocol. Thus we evaluate both the protocols on the basis of propagation time and traffic analysis. Thus we will observe that websocket proves to be a better option for video streaming over a traditional HTTP protocol.

I.INTRODUCTION

1.1 Problem Statement:

Currently HTTP protocol is used by the web to connect to the server. The request sent through HTTP protocol contains a lot of overhead on bandwidth, especially in case of real-time data. Also it does not provide the facility of client and server sending and receiving data both at the same time.[2]

In normal HTTP protocol, first the client sends connection request, the server accepts it and connection is established. After communication the connection is closed. For next communication again the client needs to send separate request to the server. This results in waste of time for establishing the connection again and again. It also increases overhead [3].

1.2 Need of WebSocket protocol:

HTTP is shown down in the scenario where bidirectional communication between the client and the server is needed. HTTP sends distinct calls for upstream notifications.

This leads to many issues:

- Different connections are to be used by the server, specifically one for sending and other for receiving data.
- There is a lot of overhead in a wire protocol, where client-to-server message is using HTTP protocol.

The solution to these problems will be using a single TCP connection. This facility is provided by websocket protocol. Websocket is combined with API; it can act as a substitute to HTTP polling mechanism form a web page at remote server. This technology can be used for variety of applications.

1.3 General Idea:

Full-duplex data transmission provides transmission of data in both directions on a signal carrier at the same time. Therefore data can be transmitted back and forth simultaneously. The basic idea for this two way communication is given in following figure (Fig 1). The figure shows exact flow of websocket connection. As HTTP uses http or https connection over TCP for web, the websocket protocol uses ws or wss for web.

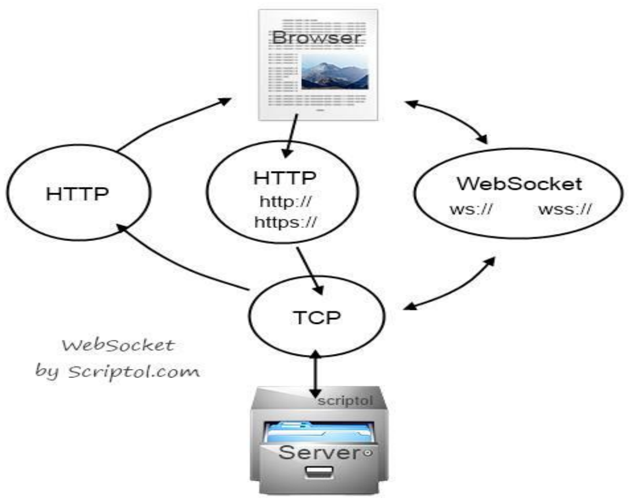

Fig (1): WebSocket Connection

II. IMPLEMENTATION

Handshaking:

Handshaking is similar to that of HTTP to provide facility of using both websocket and HTTP on the same port. After the handshake the further process is different from original HTTP protocol. The proper handshaking mechanism is explained below using a diagram (fig2) [4].

Server manages the websocket communications. Client introduces itself by sending handshake request through websocket protocol. The server replies back but sending response to the handshake. The following figure explains all the mechanisms and keys used for this process of handshaking.[5]

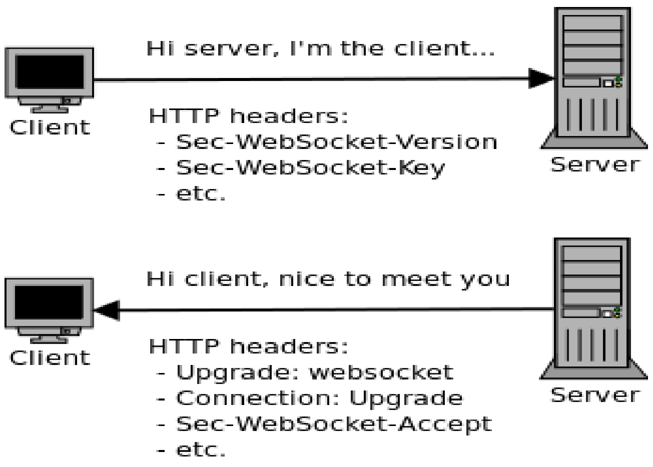

Fig (2): Opening and closing of websocket connection

Both the client and the server can communicate only after a proper connection is established. Multiple clients can be connected to the server. This works as a single server and multiple clients' connection.

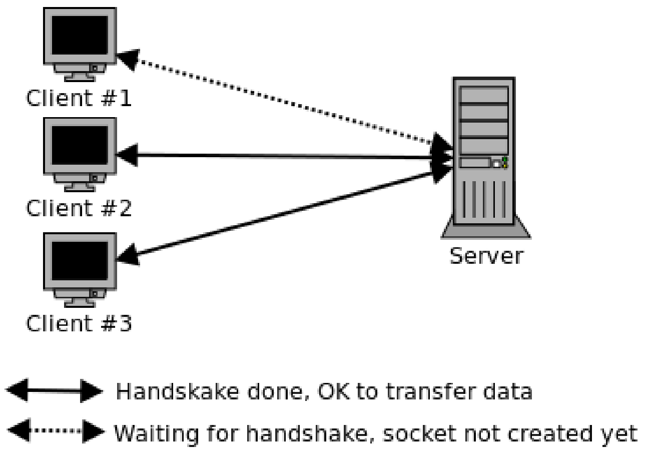
Client #1
Client #2
Client #3
Server
Handskake done, OK to transfer data
Waiting for handshake, socket not created yet

Fig (3): Communication between a Server and Multiple Clients

The customer sends a Sec-WebSocket-Key which is an arbitrary quality that has been base64 encoded. To structure a reaction, the enchantment string 258EAFA5-E914-47DA-95CA-C5AB0DC85B11 is attached to this (uudecoded) key.

The Details of the Sec-WebSocket-Key to Sec-WebSocket-Accept are mentioned below:

- x3JJHMbDL1EzLkh9GBhXDw==258EAFA5-E914-47DA-95CA-C5AB0DC85B11 string hashed by SHA-1 gives 0x1d29ab734b0c9585240069a6e4e3e91b61da1969 hexadecimal value.

- Encoding the SHA-1 hash by Base64 (utilizing the paired of the SHA-1) yields HSmrc0sMlYUkAGmm5OPpG2HaGWk=, which is the Sec-WebSocket-Accept.

After the establishment of proper connection the websocket text and data can be sent at a time i.e. in full duplex mode. The information is negligibly encircled, with a little header took after by payload. WebSocket transmissions are depicted as "messages", where a solitary message can alternatively be part over a few information outlines. This can consider sending of messages where introductory information is accessible however the complete length of the message is obscure (it sends one information outline after an alternate until the end is arrived at and checked with the FIN bit). With expansions to the convention, this can likewise be utilized for multiplexing a few streams all the while (for case to abstain from cornering utilization of an attachment for a solitary vast payload).

III. MODULES

Following modules are proposed for making the effective use.

- Message Transfering
- File Uploading
- Video streaming

3.1 Messaging Transferring:

I. Designing Client:

Client application will connect to web socket server. Messaging client will be the desktop application. This will be the start up application.

II. Designing Server:

Web socket server will manage all clients connected to it. When client connects, with server, server holds the context object of client which is latter used by server for the communication with client.

III. Messaging:

A web socket channel is used in sending, message from one client to other. The WebSocket message does less contrast with a particular framework layer encompassing, as an isolated message may be part by a center individual. All the frames have their own types.

There are sorts for literary information, binary data and control outlines. WebSocket association foundation model, WebSocket convention is basically a TCP. To create a Web Socket association, the customer and server update.

3.2 File Transferring:

Client application can also send file to other clients via web socket. Files can also be sent to multiple clients and the server can also receive files from multiple clients with authentication. Different clients have to register with the server for transaction of files.

3.3 Streaming Server:

I. Designing Web Client:

This client will be the web application listening the list of the videos uploaded by administrator.

II. Video Management:

This module is responsible for the management of all uploaded videos like adding new video, deleting videos.

Only the authenticated client can perform altering with the videos.

III. Video Upload:

Admin or user can upload videos from this module. Video uploading speed is very fast because of the use of websocket protocol.

IV. PERFORMANCE EVALUATION

The assessment of the websocket and TCP convention consists of comparing both of them on the basis of system performance and the time required to exchange information between client and server. As the limit of information exchange in both protocols depends solely on payload information, we can assess their efficiency in a logical manner. After this the information exchange time is also evaluated. Network traffic also comes in consideration of this assessment when internet is involved.[1]

Hence, our research is related to comparison of TCP and websocket protocol on the basis of overhead. We can say that this overhead is nothing but the message exchange amongst the websocket handshake and it also outlines every header used for such exchange. Several bytes are checked and the handshake is settled long for the overhead of websocket handshake. [6] We must concentrate on performance of long running sessions

The above graph shows that the TCP handshake duration is for about 965 ms, whereas websocket handshake duration is 3533 ms. Which shows that a websocket connection lasts for longer time and hence we can communicate longer.

.

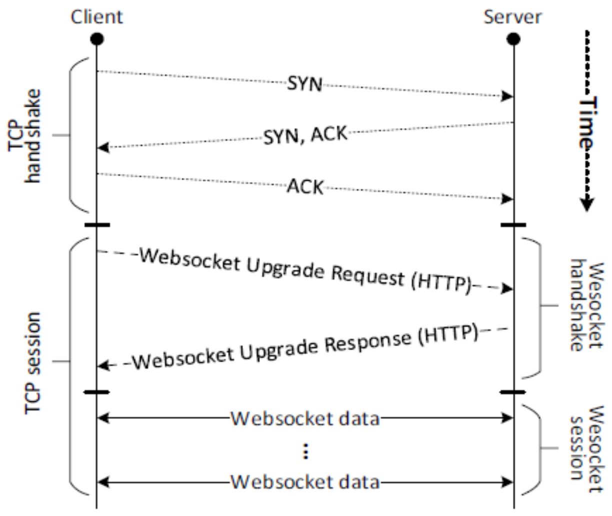

Fig (4) Websocket vs. TCP sequence diagram

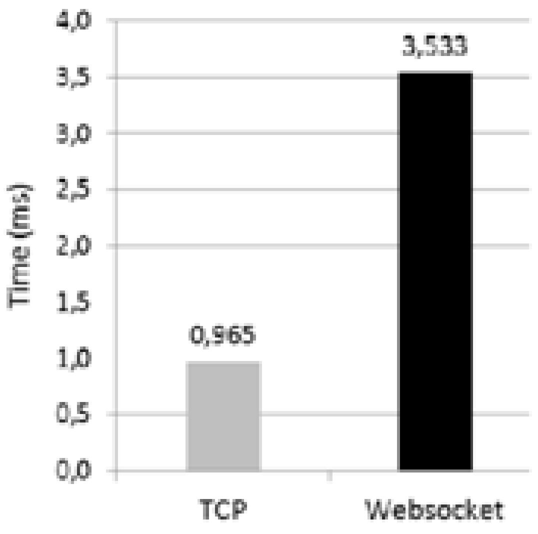

Fig (5) Comparison of TCP and Websocket handshake duration

V. DISCUSSION

The client can get to the feature quick. The time needed for extricating the feature is less. To play the feature, each time it is not important to download the glimmer player. User can select streaming type secure or a non-secure. Association built in the middle of customer and server with the assistance of WebSocket will be a dependable association that implies until and unless we stop the server or close the program window association ways out and correspondence can move ahead. As we are using full-duplex network, there will be only one stream through which request and response occurs simultaneously so overhead is less.

VI. CONCLUSION

The websocket protocol provides bidirectional communication in a world where most of the web processing is based on the client-server model. In this, client is majorly responsible for starting the request/response pairs. Even though some of the limitations of the TCP socket like communication capabilities are overcome by websocket protocol related to HTTP based applications, a little performance issue of websocket compared to conventional TCP protocol should not be considered. For long time running web sessions in which data is streams continuously between client and server, the payload matters a lot.

Websocket Protocol enables fast streaming than traditional video streaming approach in a secured manner consuming low bandwidth. Thus we can conclude that the main advantage of using websocket protocol for video streaming is that it uses single socket to transfer information between the client and the server. This will not only provide low bandwidth overload but also will increase portability, high efficiency and smooth video streaming.

REFERENCES

1. Skvorc, D., M. Horvat, and S. Srbljic. Performance evaluation of Websocket protocol for implementation of full-duplex web streams. in Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on. 2014.

2. Pimentel, V. and B.G. Nickerson, Communicating and Displaying Real-Time Data with WebSocket. Internet Computing, IEEE, 2012. **16**(4): p. 45-53.

3. Cunha, C.A. and L. Moura e Silva. SHStream: Self-Healing Framework for HTTP Video-Streaming. in Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on. 2013.

4. Wessels, A., et al. Remote Data Visualization through WebSockets. in Information Technology: New Generations (ITNG), 2011 Eighth International Conference on. 2011.

5. Nakajima, H., M. Isshiki, and Y. Takefuji. WebSocket proxy system for mobile devices. in Consumer Electronics (GCCE), 2013 IEEE 2nd Global Conference on. 2013.

6. Watanabe, T., et al. A low latency and intuitive control video streaming system. in Consumer Electronics (GCCE), 2012 IEEE 1st Global Conference on. 2012.